Sri Lanka Institute of Information Technology

Assignment 1

Data Warehouse and Business Intelligence

2022

Submitted by:

Hasna A K F

IT20212940

Contents

Data set Selection and Preparation

The selected data source is a collection of transaction data. This data set reflects donation transactions done by donors. Donors' specific details involved in transactions, Details of schools which receive donations, Rating done by schools, Details of teachers who post projects, Project Details are the key details included in the data set.

The two main sources are listed below:

SQL Database

One text file – Teachers Data

The csv files that were imported to the SQL database are listed below:

Donors Data

Donations Data

Schools Data

Projects Data

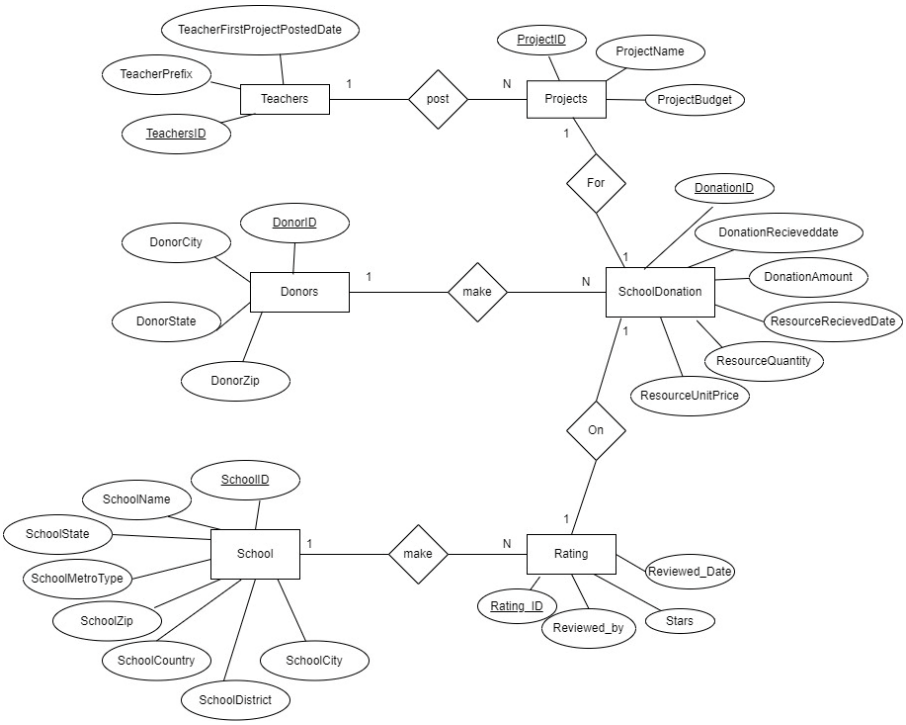
Rating Data

Description of the Data Set:

Table name	**Column name**	**Data type**	**Description**
Donors	DonorID	nvarchar(255)	Donor Details
	DonorCity	nvarchar(255)	
	DonorState	nvarchar(255)	
	DonorZip	float	
Projects	ProjectID	nvarchar(255)	Project Details
	TeachersID	nvarchar(255)	
	ProjectName	nvarchar(255)	
	ProjectBudget	float	
Rating	Rating_ID	nvarchar(255)	Details of Rating by Schools
	School_ID	nvarchar(255)	
	Revieved_by	nvarchar(255)	
	Stars	numeric(38,0)	
	Reviewed_Date	datatime	

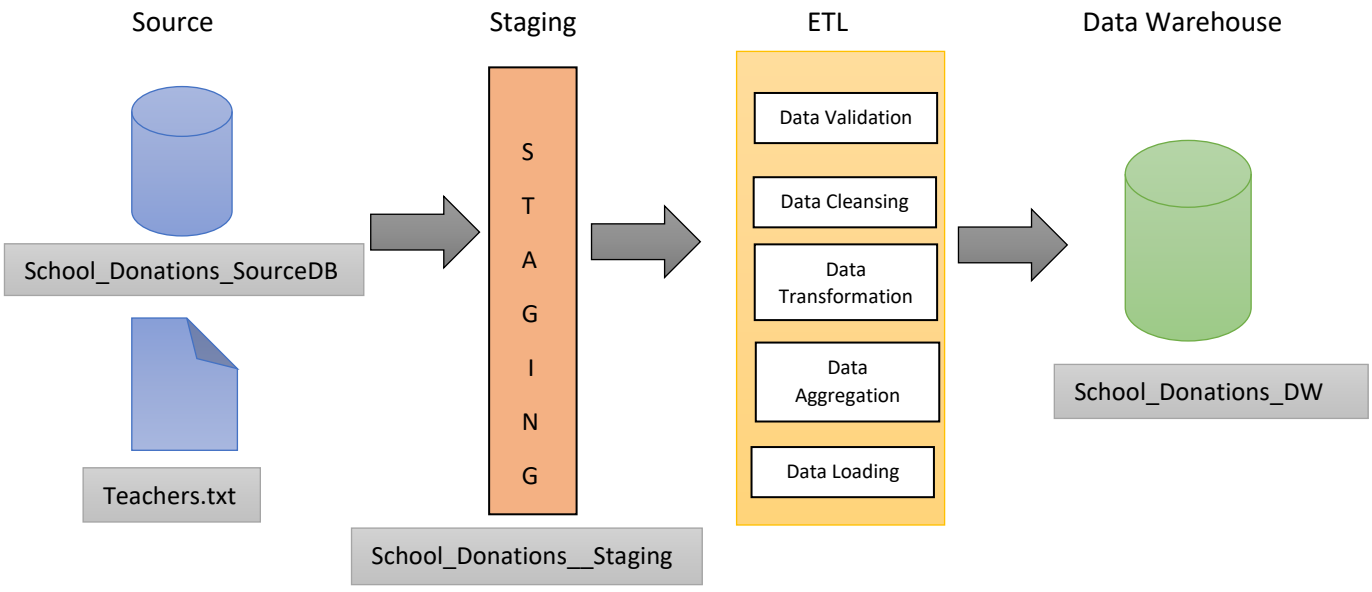
SchoolDonation	DonationID	nvarchar(255)	Details of Donation Transactions
	DonationRecievedDate	datetime	
	DonationAmount	float	
	ProjectID	nvarchar(255)	
	ResourceQuantity	float	
	ResourceUnitPrice	float	
	ResourceRecievedDate	datetime	
	RatingID	nvarchar(255)	
	DonorID	nvarchar(255)	
Schools	SchoolID	nvarchar(255)	School Details
	SchoolName	nvarchar(255)	
	SchoolMetroType	nvarchar(255)	
	SchoolState	nvarchar(255)	
	SchoolZip	float	
	SchoolCity	nvarchar(255)	
	SchoolCountry	nvarchar(255)	
	SchoolDistrict	nvarchar(255)	
Teachers	TeacherID	varchar(70)	Teachers Details
	TeacherPrefix	varchar(50)	
	TeacherFirstProjectPostedDate	datetime	

ER Diagram



This diagram elaborates the connection between the entities in the data set

Solution Architecture



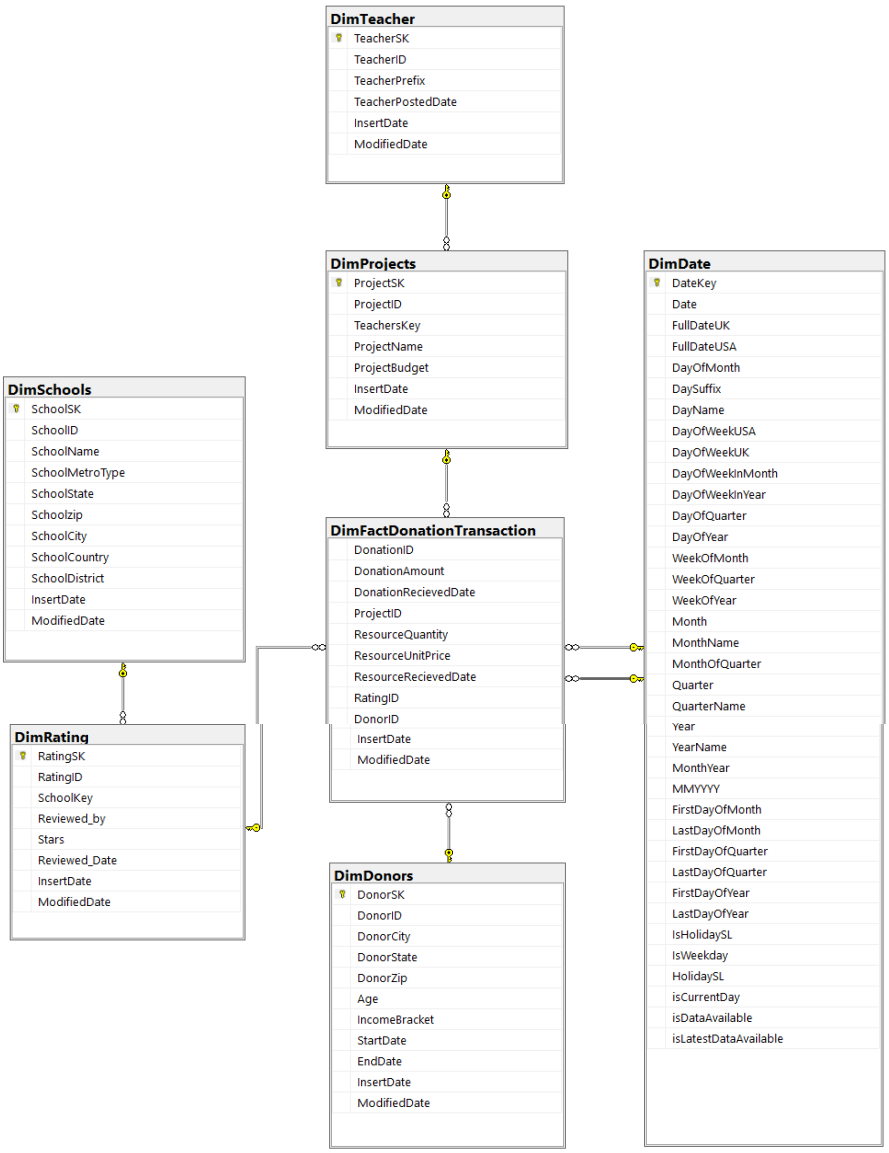
The first step is to stage the source data set. After the staging process the below listed tables were created:

1. stg_Donations
2. stg_Donors
3. stg_Projects
4. stg_Rating
5. stg_Schools
6. stg_Teachers

In the next the staged tables are profiled. As the next step the data is transformed and loaded. After completing the above mentioned steps data is tested and validated and finally the data warehouse is created.

After creating the data warehouse BI results such as OLAP analysis, Data mining, Data Visualization, Reports can be obtained.

Data Warehouse Design and Development



The Database is designed using Snowflake schema. There is one fact table named DimFactDonations and five dimensions named DimRating, DimSchools, DimDonors, DimProjects and DimTeachers.

Assumptions.

Donor dimension is considered as a slowly changing dimension.

Test Planning and Test Data

Testing is done to ensure that the data is accurate when it is loaded from source to the destination. Testing involves verification of data at various middle stages throughout the transformation from source to destination.

In this project data is tested in two stages as mentioned below:

1. Source to Staging (Middle Stage)
2. Staging to Dimension tables (Final Destination)

Test Plan

Scope	1. Completeness of the data set testing- These test cases are conducted to ensure that there is no loss of data after it is loaded. 2. Data length testing- To ensure that the data length matches when the data is loaded from source to middle stages and then to the final destination. 3. Data type testing- Data types are tested in order to avoid data type mismatch issues along the process. 4. Data Duplicity testing- To ensure that the data is not getting duplicated in order to maintain the quality
Out of Scope	Validity of data testing
Assumptions	There is no environment downtime during testing
Schedules	Start Date- 25/04/2022 End Date- 10/05/2022
Roles and responsibilities	A.K.F Hasna 1. Create Test plan 2. Generate Test Cases 3. Execute Test Cases 4. Create test summary report
Test Deliverables	1. Test Plan

	2. Test Cases and Results 3. Test summary report
Test Environment	Database Server: SQL Server Management Studio Operating System: Windows 10
Test tools	Microsoft SQL Server Data Tools for Visual Studio 2019

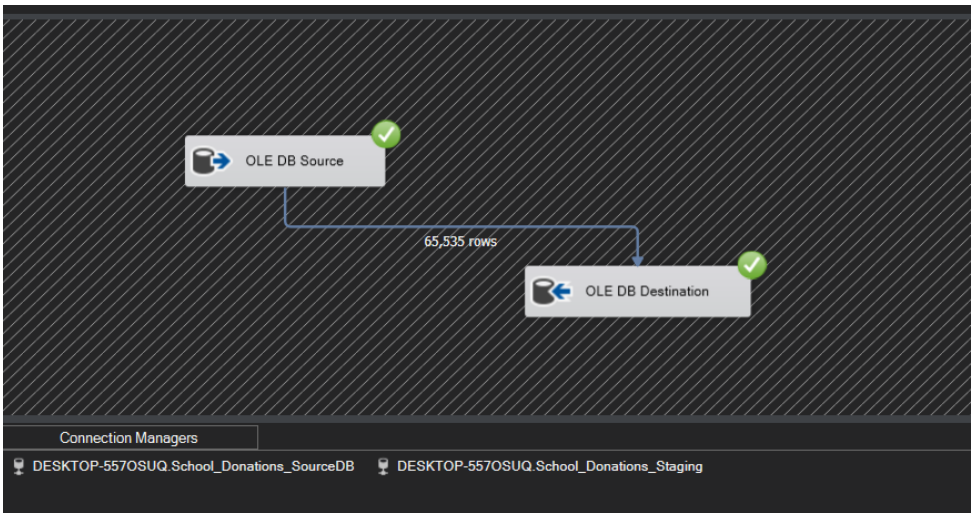
ETL Development

As the Initial step data was extracted from the sources (DB source and text file). Every data source was extracted to the staging using a data flow task. Every staging table was truncated in order to avoid data duplicates. The process of loading data from source to staging is described below.

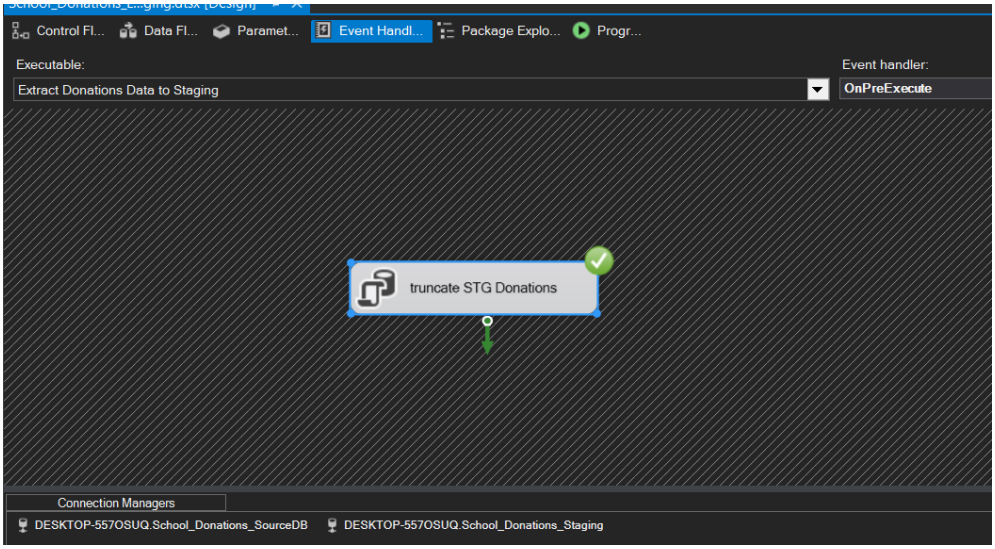
Screenshots of all data sources that were staged and truncate tables created as attached below.

Staging

Staging Donation Details.

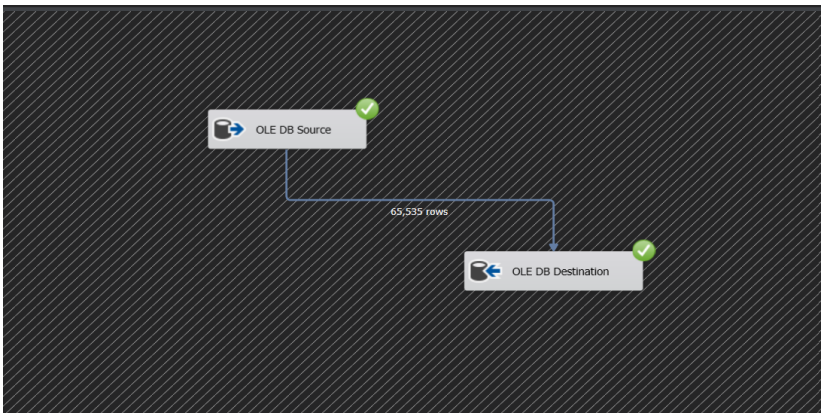


The above image illustrates the extraction of data from Donations table (Source) into Donations Staging Table

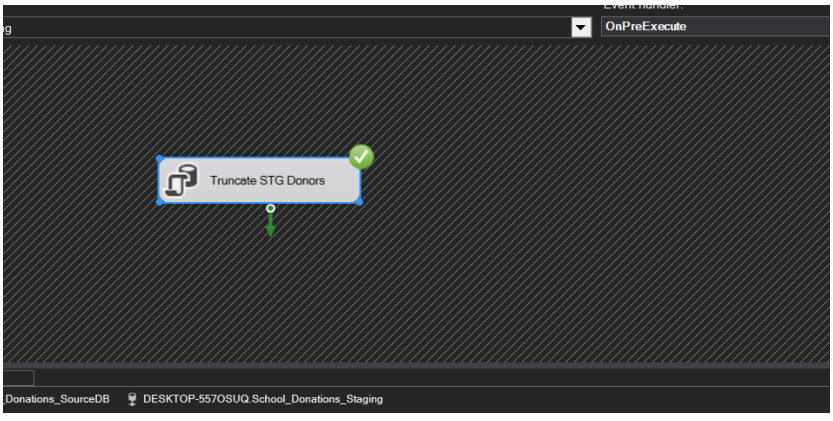


The above image illustrates the creation of Donations truncate table in order to avoid duplicate data.

Staging Donors Details.

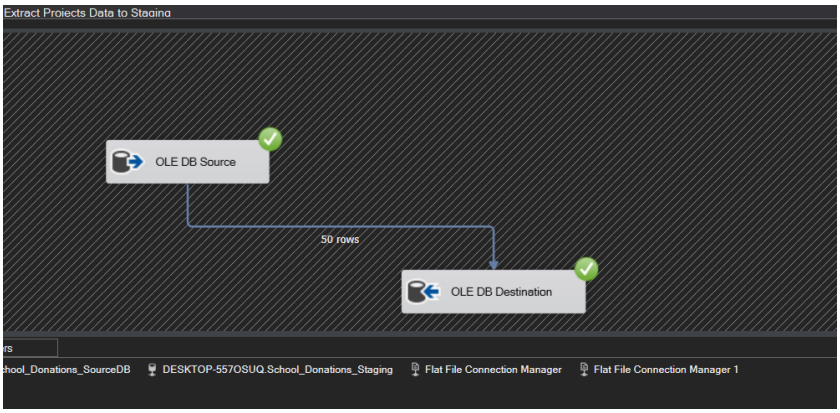


The above image illustrates the extraction of data from Donors table (Source) into Donors Staging Table

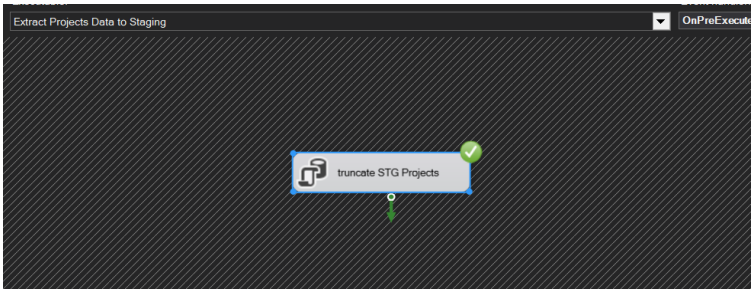


The above image illustrates the creation of Donors truncate table in order to avoid duplicate data.

Staging Projects Details.

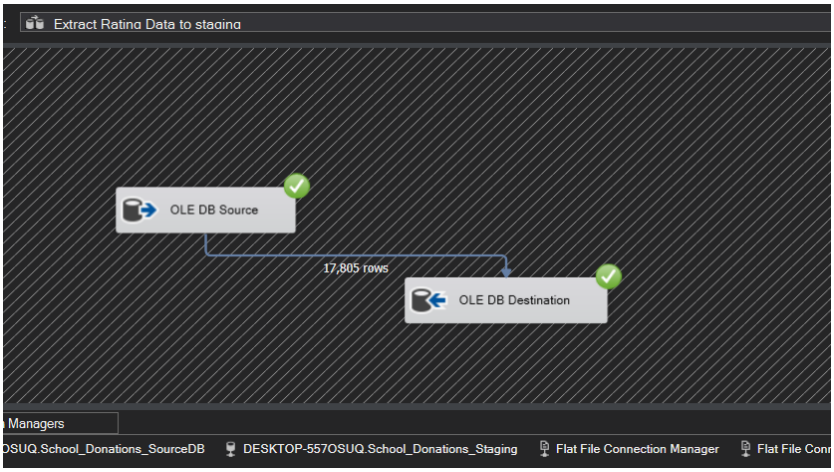


The above image illustrates the extraction of data from Projects table (Source) into Projects Staging Table

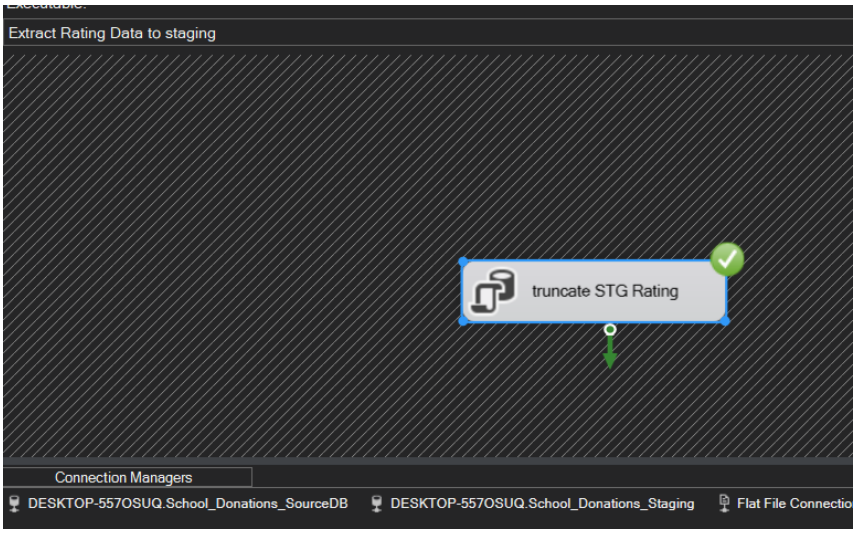


The above image illustrates the creation of Projects truncate table in order to avoid duplicate data.

Staging Rating Details.

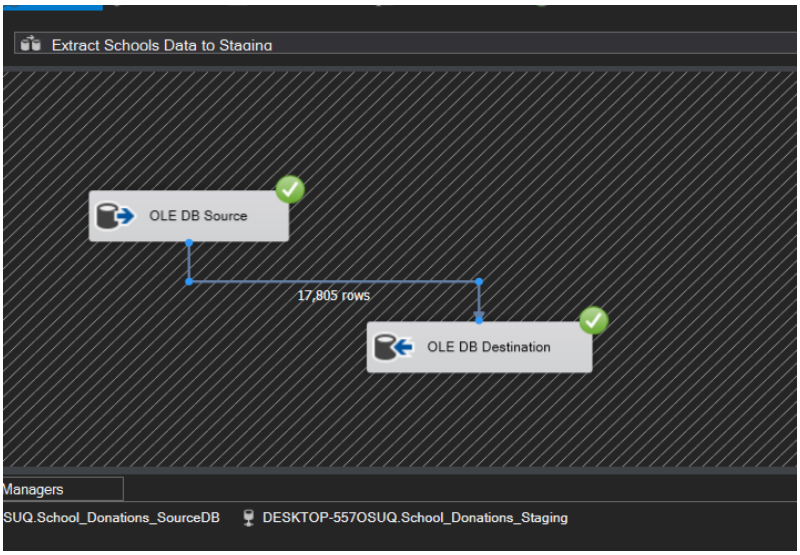


The above image illustrates the extraction of data from Rating table (Source) into Rating Staging Table

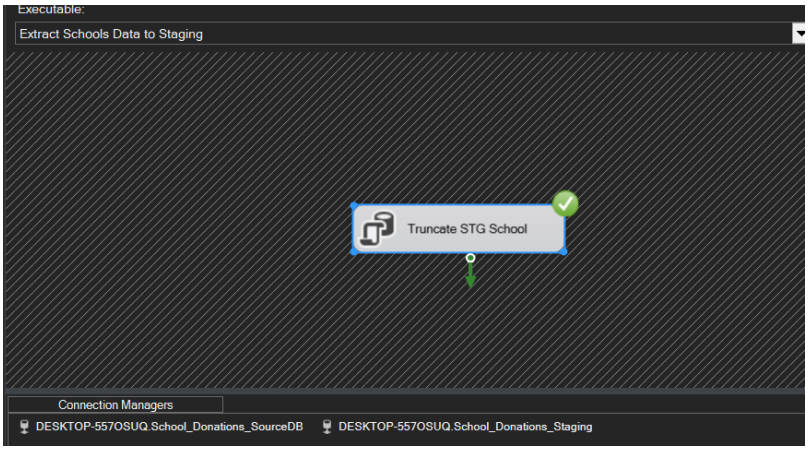


The above image illustrates the creation of Rating truncate table in order to avoid duplicate data.

Staging Schools Details.

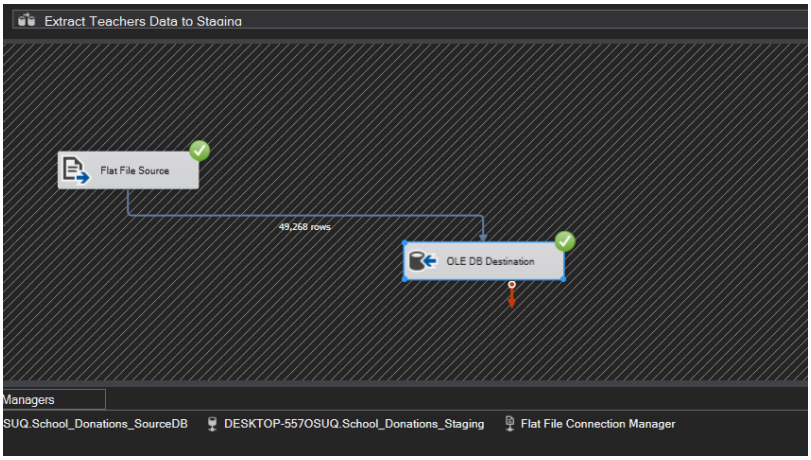


The above image illustrates the extraction of data from Schools table (Source) into Schools Staging Table

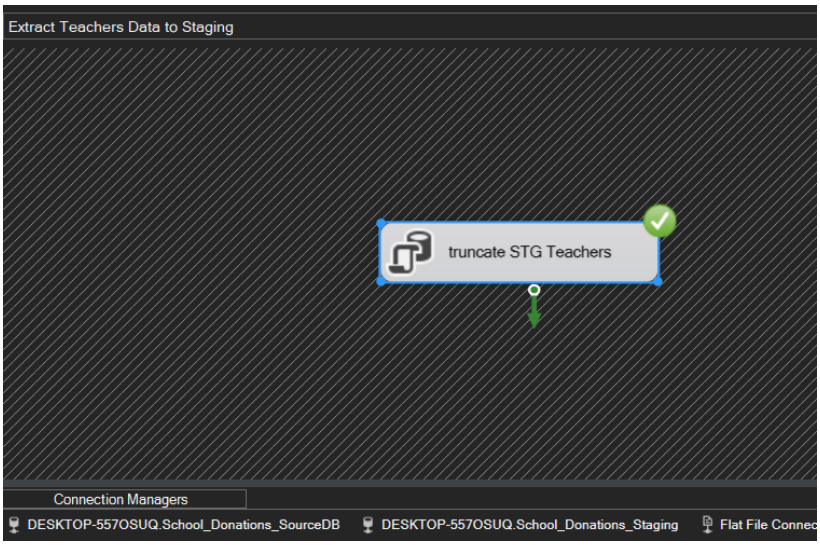


The above image illustrates the creation of Schools truncate table in order to avoid duplicate data.

Staging Teachers Details.

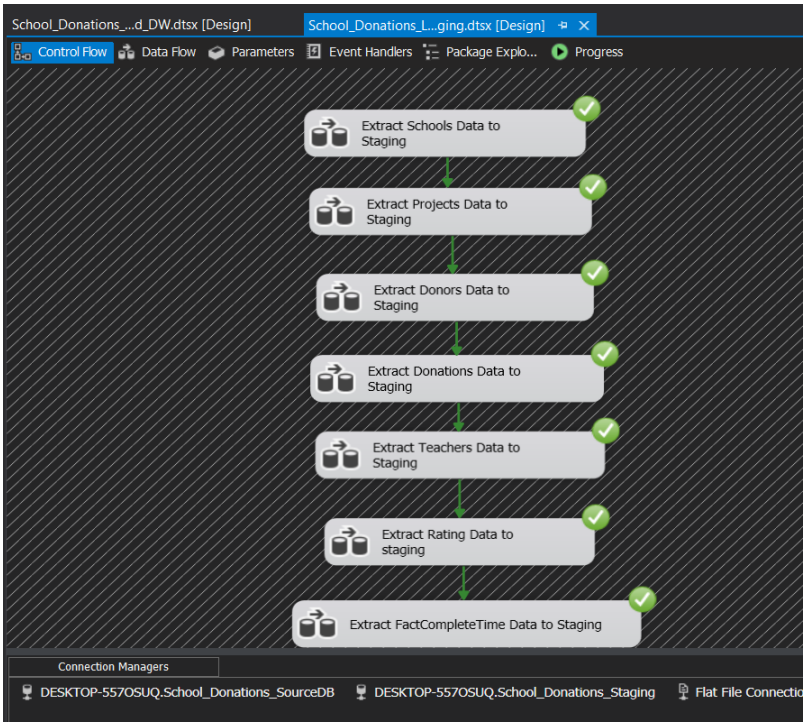


The above image illustrates the extraction of data from Teachers table (Text file) into Teachers Staging Table



The above image illustrates the creation of Teachers truncate table in order to avoid duplicate data.

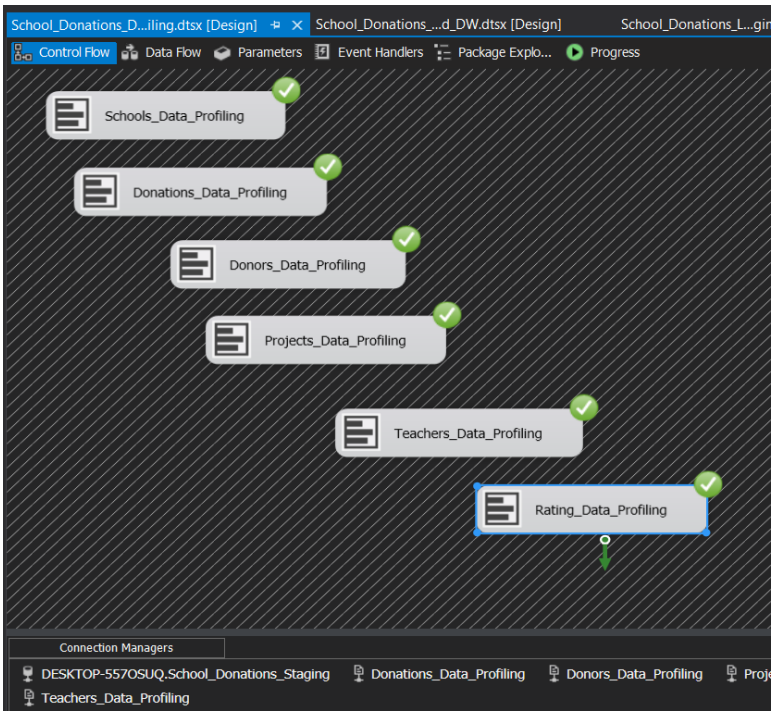
After extracting all the data sources to staging tables.



The above image illustrates the complete execution of the Staging package

Data Profiling

Next step is Data Profiling



Every Staging table is profiled and saved in a particular location

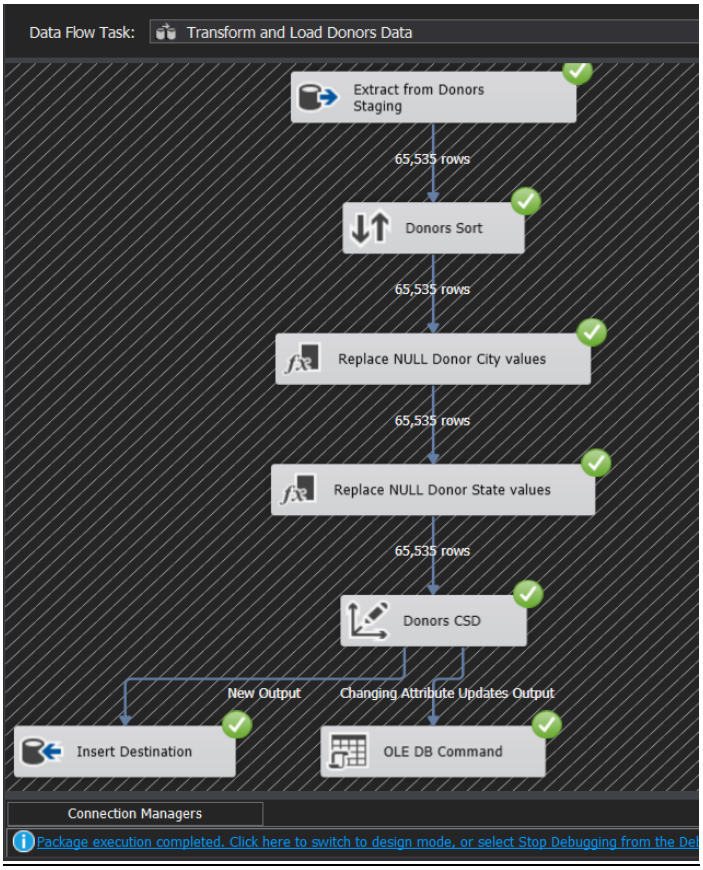
Data Transformation

The next step is data transformation which is loading data from the staging tables into dimension tables.

As mentioned under assumptions, earlier Donor details were considered as slowly changing details

1. Donor city (City in which donor resides)
2. Donor Zip
3. Age
4. Income Bracket

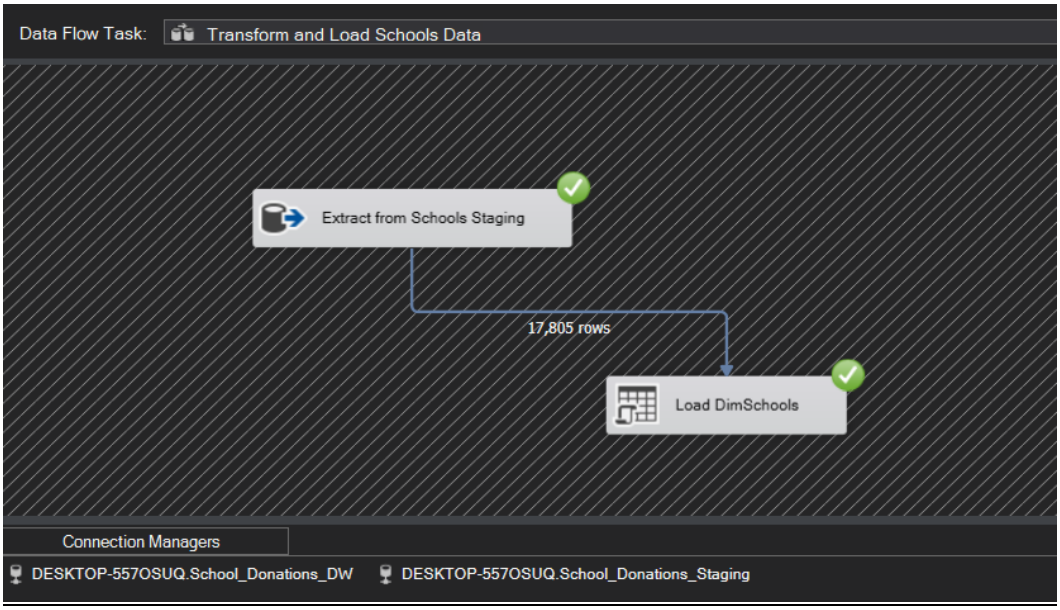
Transform and Load Donors Data



The above image illustrates that the data is extracted from Donors Staging Table and loaded into Donors Dimension

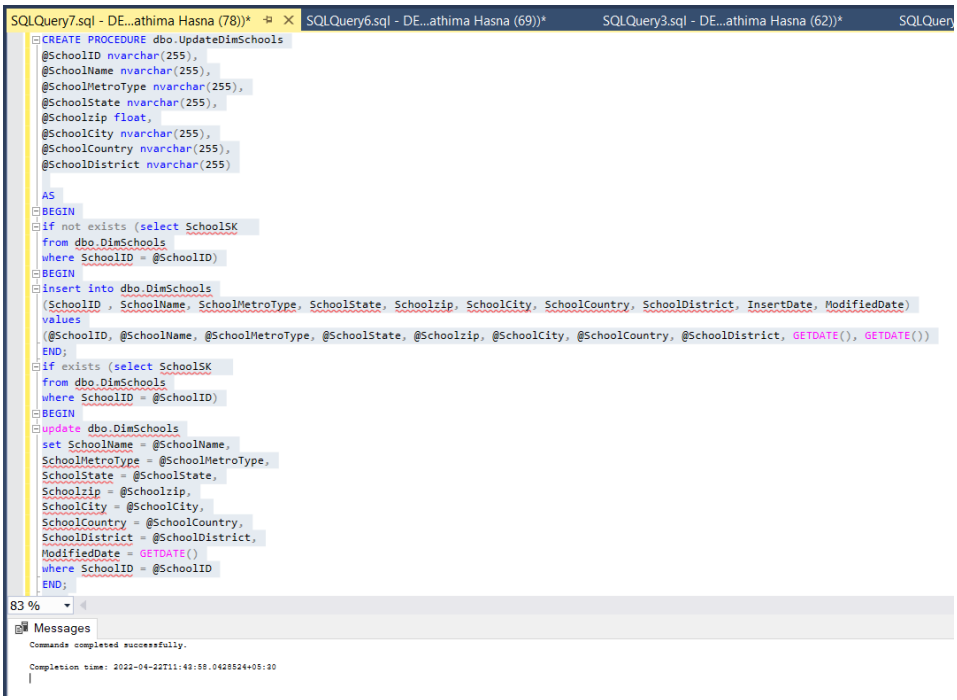
First the data is extracted from the Donors staging table was sorted according to the donor id, then it was considered as a slowly changing dimension. Finally, the data was loaded into the Donors Dimension table.

Transform and Load Schools Data



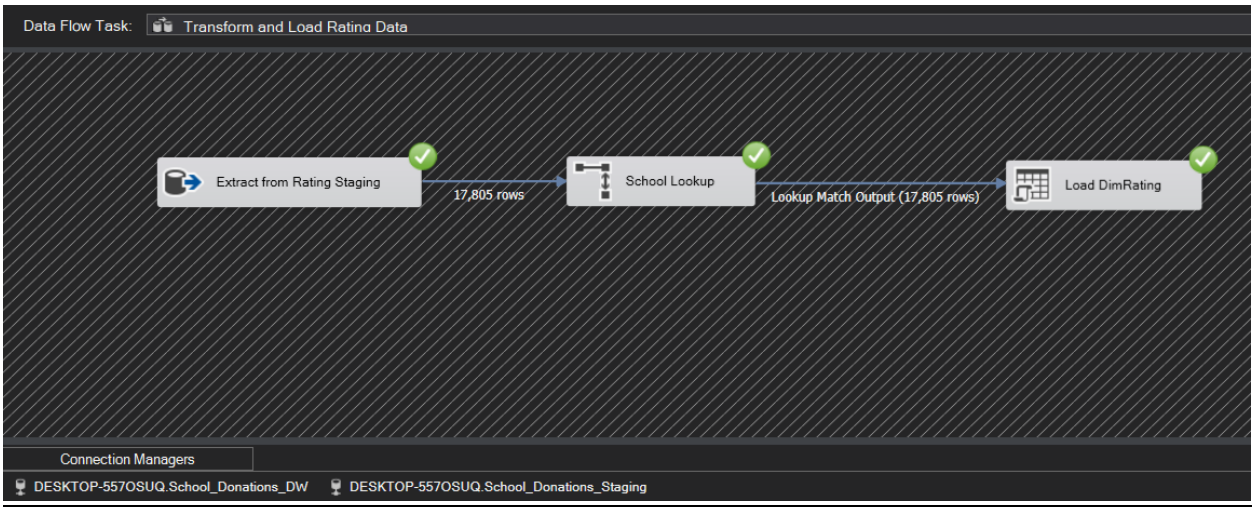
The above image illustrates that the data is extracted from Schools Staging Table and loaded into Schools Dimension

The update procedure used to update Schools details is attached below:



```
SQLQuery7.sql - DE...athima Hasna (78))*   SQLQuery6.sql - DE...athima Hasna (69))*   SQLQuery3.sql - DE...athima Hasna (62))*   SQLQuery

CREATE PROCEDURE dbo.UpdateDimSchools
@SchoolID nvarchar(255),
@SchoolName nvarchar(255),
@SchoolMetroType nvarchar(255),
@SchoolState nvarchar(255),
@Schoolzip float,
@SchoolCity nvarchar(255),
@SchoolCountry nvarchar(255),
@SchoolDistrict nvarchar(255)

AS
BEGIN
if not exists (select SchoolSK
from dbo.DimSchools
where SchoolID = @SchoolID)
BEGIN
insert into dbo.DimSchools
(SchoolID , SchoolName, SchoolMetroType, SchoolState, Schoolzip, SchoolCity, SchoolCountry, SchoolDistrict, InsertDate, ModifiedDate)
values
(@SchoolID, @SchoolName, @SchoolMetroType, @SchoolState, @Schoolzip, @SchoolCity, @SchoolCountry, @SchoolDistrict, GETDATE(), GETDATE())
END;
if exists (select SchoolSK
from dbo.DimSchools
where SchoolID = @SchoolID)
BEGIN
update dbo.DimSchools
set SchoolName = @SchoolName,
SchoolMetroType = @SchoolMetroType,
SchoolState = @SchoolState,
Schoolzip = @Schoolzip,
SchoolCity = @SchoolCity,
SchoolCountry = @SchoolCountry,
SchoolDistrict = @SchoolDistrict,
ModifiedDate = GETDATE()
where SchoolID = @SchoolID
END;

83 %
Messages
Commands completed successfully.

Completion time: 2022-04-22T11:43:58.0428524+05:30
```

Transform and Load Rating Data



The above image illustrates that the data is extracted from Rating Staging Table and loaded into Rating Dimension

First the data is extracted from the Rating staging and School ID is looked up from the Schools Dimension and school surrogate key is inserted into the Rating Dimension as a foreign key and finally loaded the data into the Rating Dimension

The update procedure used to update rating details is attached below:

SQLQuery1.sql - DE...athima Hasna (56))*

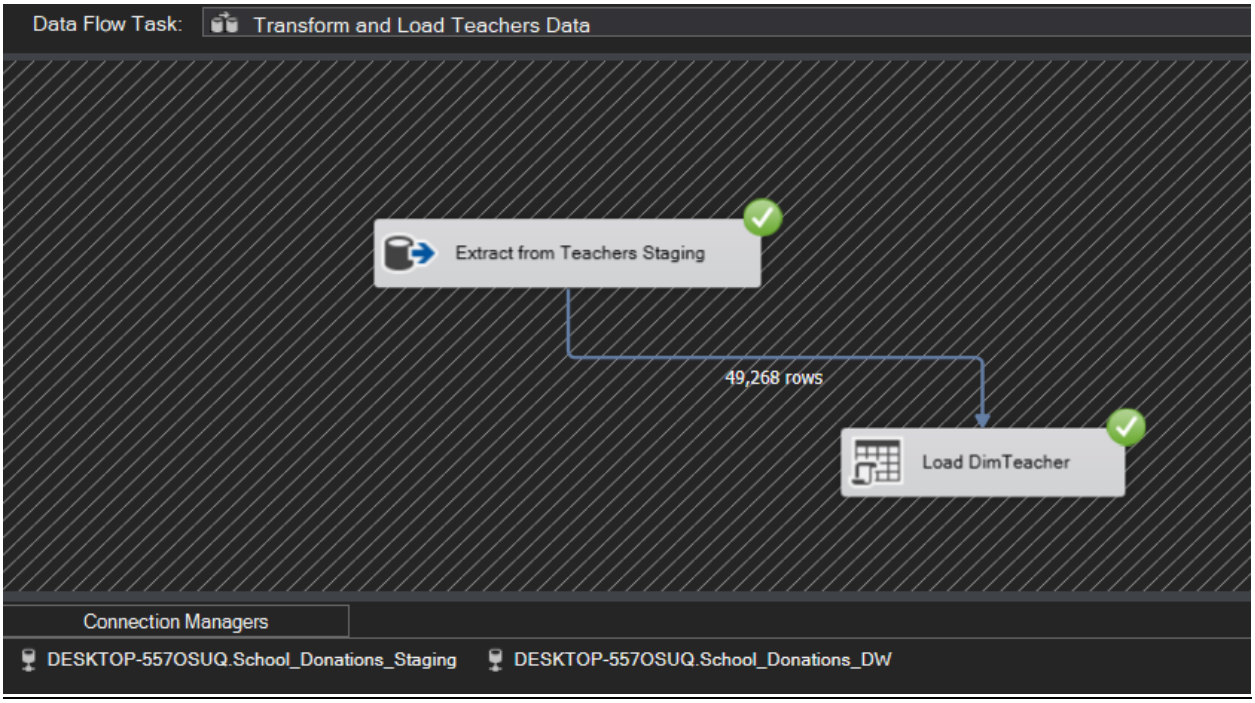
```
CREATE PROCEDURE dbo.UpdateDimRating
@RatingID  nvarchar(255),
@SchoolKey int,
@Reviewed_by nvarchar(255),
@Stars numeric(38,0),
@Reviewed_Date  DateTime
AS
BEGIN
if not exists (select RatingSK
from dbo.DimRating
where RatingID = @RatingID)
BEGIN
insert into dbo.DimRating
(RatingID , SchoolKey, Reviewed_by, Stars, Reviewed_Date,
InsertDate, ModifiedDate)
values
(@RatingID, @SchoolKey, @Reviewed_by, @Stars, @Reviewed_Date,
GETDATE(), GETDATE())
END;
if exists (select RatingSK
from dbo.DimRating
where RatingID = @RatingID)
BEGIN
update dbo.DimRating
set SchoolKey   = @SchoolKey,
Reviewed_by = @Reviewed_by,
Stars = @Stars,
Reviewed_Date = @Reviewed_Date,
ModifiedDate = GETDATE()
where RatingID = @RatingID
END;
```

100 %

Messages

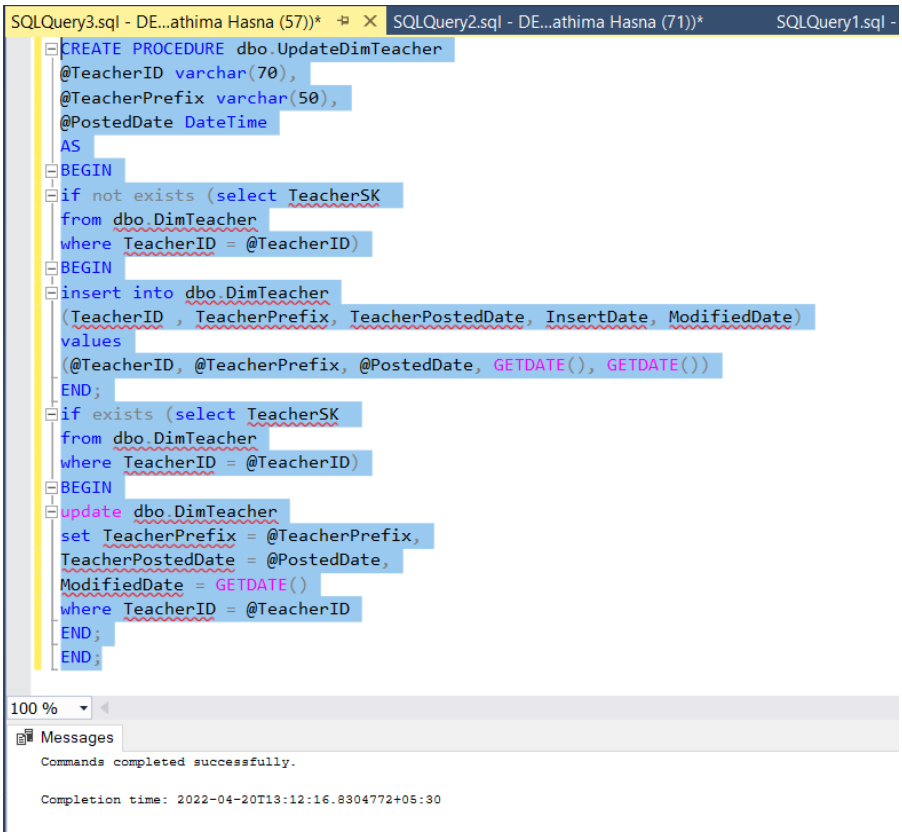
```
Commands completed successfully.

Completion time: 2022-04-25T15:09:35.7764384+05:30
```

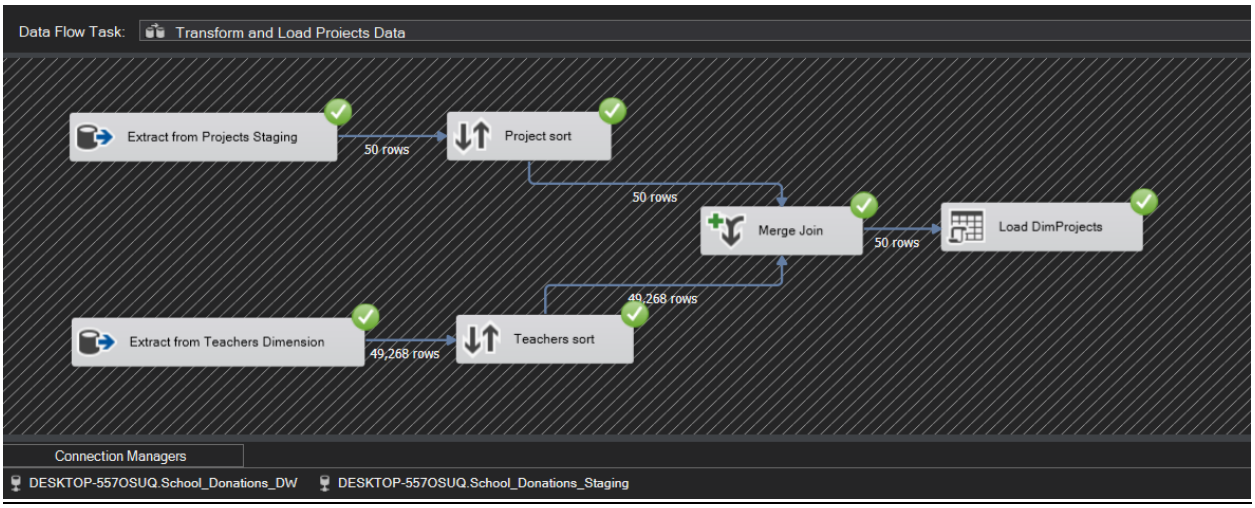
Transform and Load Teachers Data



The above image illustrates that the data is extracted from Teachers Staging Table and loaded into Teachers Dimension

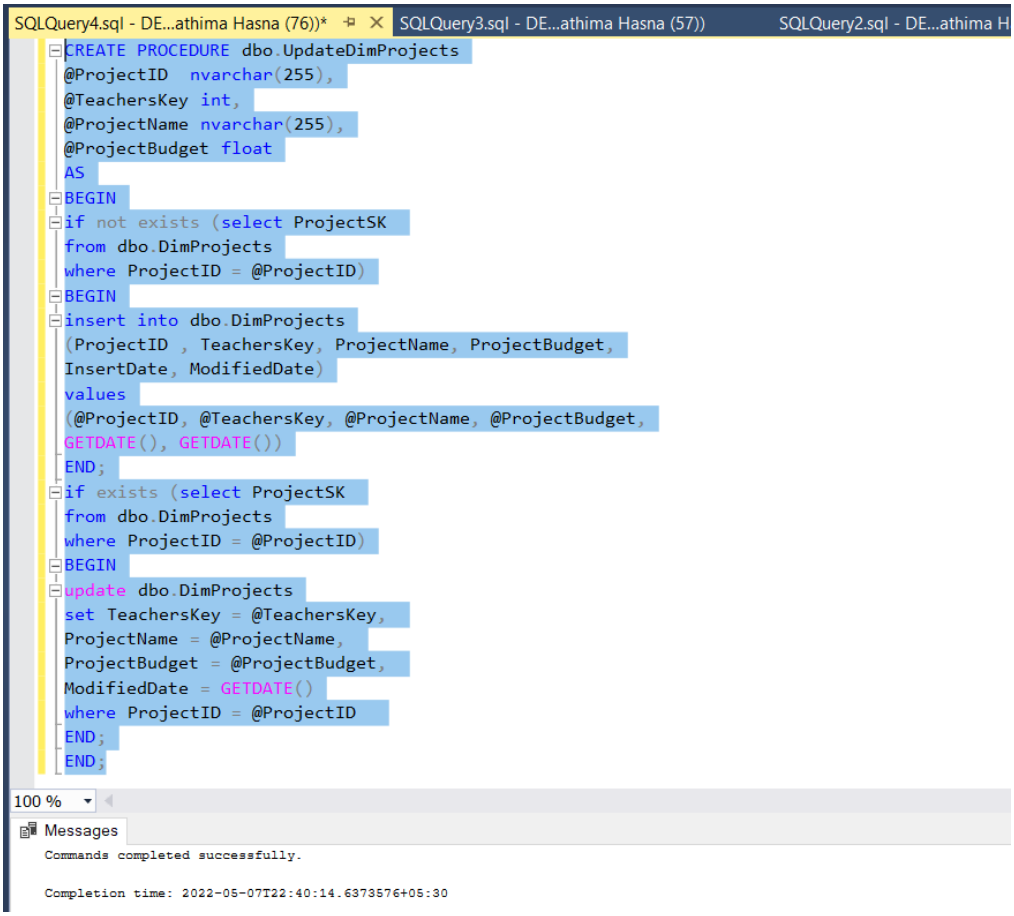
The update procedure used to update teachers details is attached below:



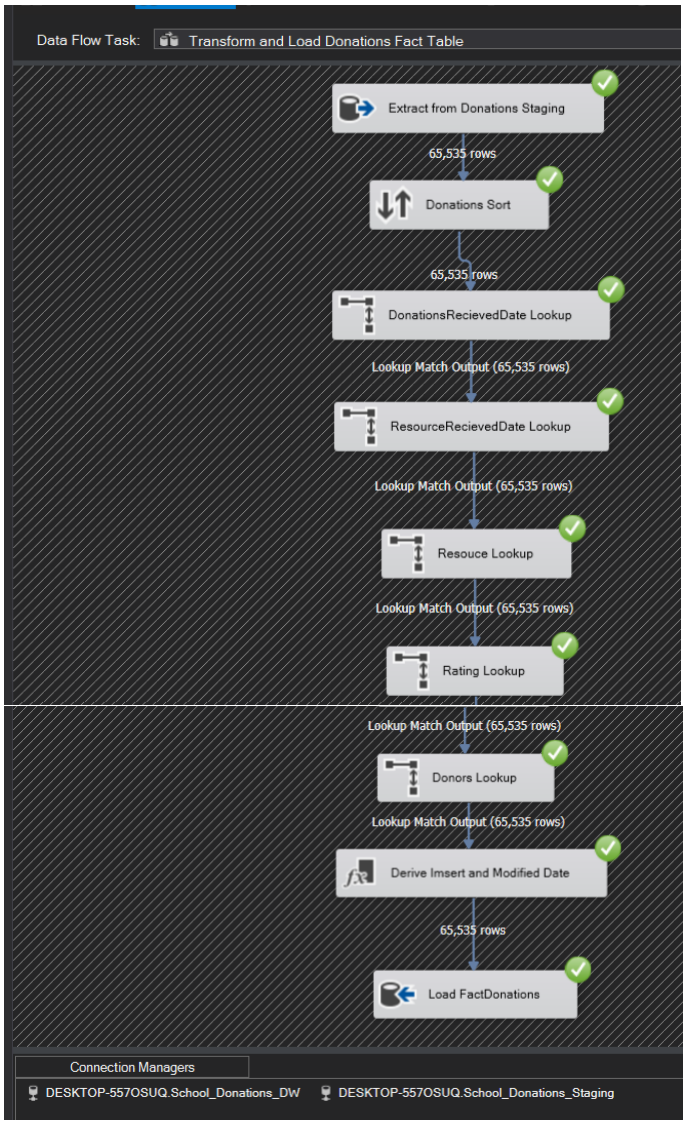


First the data is extracted from the Projects staging was sorted according to the projects Id, then the data was extracted from the Teachers staging was sorted according to the teachers Id, then the projects data and the teachers surrogate key are merged and loaded into the projects dimension.

The update procedure used to update projects details is attached below:



Transform and Load Donations Fact Table Data



After loading all the staging tables into the relevant dimensions, the fact table was loaded by following the steps mentioned below:

1. Data was extracted from donations staging
2. Sort Donations according to the donations ID
3. Map the Donation Received date with the surrogate key of the date dimension using a lookup
4. Map the Resource Received date with the surrogate key of the date dimension using a lookup
5. Map the Rating ID with the surrogate key of the rating dimension using a lookup
6. Map the Donors ID with the surrogate key of the donors dimension using a lookup

7. Derive Insert and Modified date values using the GETDATE() function
8. The date is loaded into the FactDonations Dimension

The query which was used to generate the data dimension is mentioned below:

```
CREATE TABLE    [dbo].[DimDate]
        (       [DateKey] INT primary key,
                [Date] DATETIME,
                [FullDateUK] CHAR(10), -- Date in dd-MM-yyyy format
                [FullDateUSA] CHAR(10),-- Date in MM-dd-yyyy format
                [DayOfMonth] VARCHAR(2), -- Field will hold day number of Month
                [DaySuffix] VARCHAR(4), -- Apply suffix as 1st, 2nd ,3rd etc
                [DayName] VARCHAR(9), -- Contains name of the day, Sunday, Monday
                [DayOfWeekUSA] CHAR(1),-- First Day Sunday=1 and Saturday=7
                [DayOfWeekUK] CHAR(1),-- First Day Monday=1 and Sunday=7
                [DayOfWeekInMonth] VARCHAR(2), --1st Monday or 2nd Monday in Month
                [DayOfWeekInYear] VARCHAR(2),
                [DayOfQuarter] VARCHAR(3),
                [DayOfYear] VARCHAR(3),
                [WeekOfMonth] VARCHAR(1),-- Week Number of Month
                [WeekOfQuarter] VARCHAR(2), --Week Number of the Quarter
                [WeekOfYear] VARCHAR(2),--Week Number of the Year
                [Month] VARCHAR(2), --Number of the Month 1 to 12
                [MonthName] VARCHAR(9),--January, February etc
                [MonthOfQuarter] VARCHAR(2),-- Month Number belongs to Quarter
                [Quarter] CHAR(1),
                [QuarterName] VARCHAR(9),--First,Second..
                [Year] CHAR(4),-- Year value of Date stored in Row
                [YearName] CHAR(7), --CY 2012,CY 2013
                [MonthYear] CHAR(10), --Jan-2013,Feb-2013
                [MMYYYY] CHAR(6),
                [FirstDayOfMonth] DATE,
                [LastDayOfMonth] DATE,
                [FirstDayOfQuarter] DATE,
                [LastDayOfQuarter] DATE,
                [FirstDayOfYear] DATE,
                [LastDayOfYear] DATE,
                [IsHolidaySL] BIT,-- Flag 1=National Holiday, 0-No National Holiday
                [IsWeekday] BIT,-- 0=Week End ,1=Week Day
                [HolidaySL] VARCHAR(50),--Name of Holiday in US
                [isCurrentDay] int, -- Current day=1 else = 0
                [isDataAvailable] int, -- data available for the day = 1, no data available for the day = 0
                [isLatestDataAvailable] int
        )
GO


/*****************************************************************************************
**/
--Specify Start Date and End date here
--Value of Start Date Must be Less than Your End Date
```

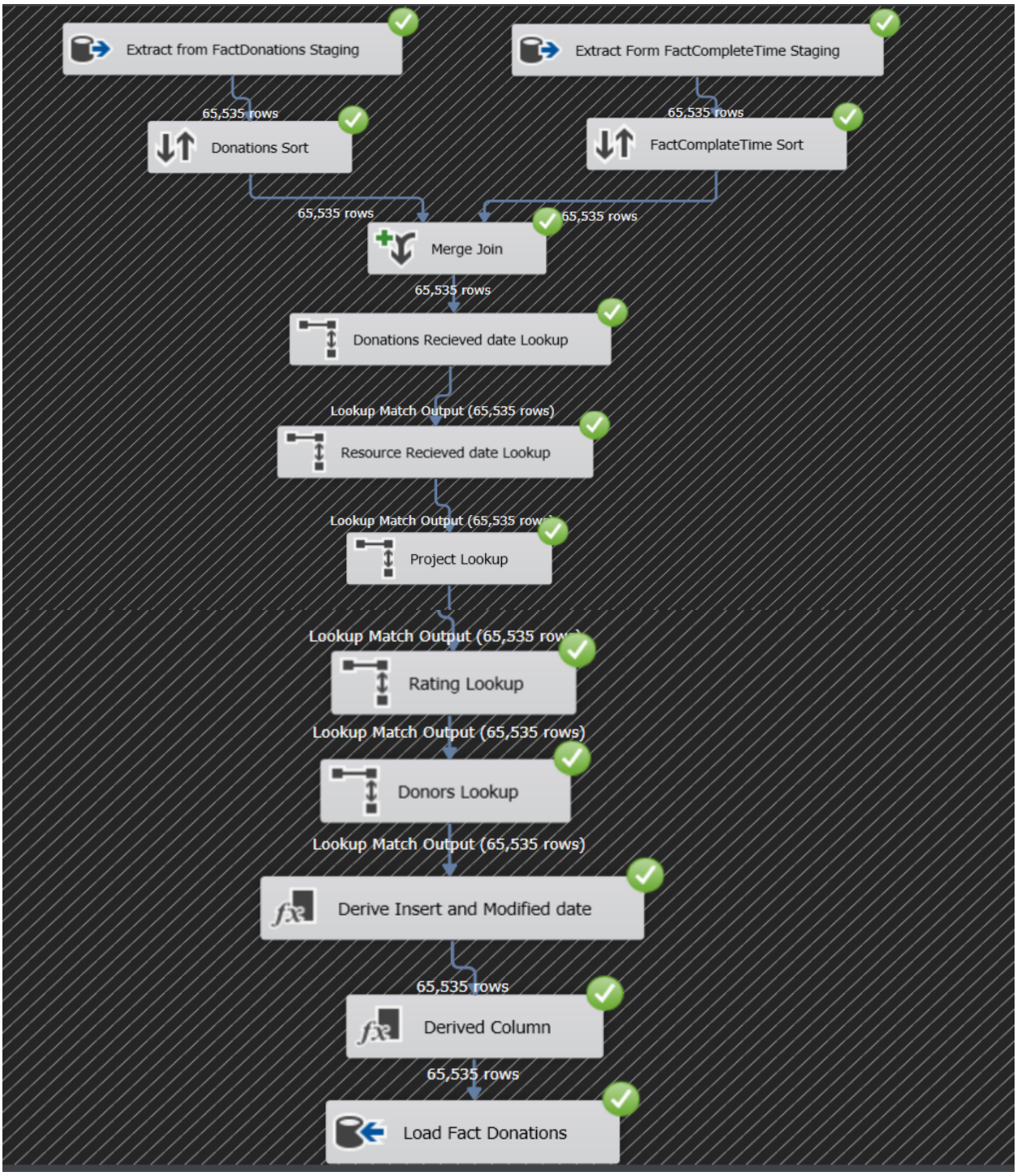
```
DECLARE @StartDate DATETIME = '01/01/1990' --Starting value of Date Range
DECLARE @EndDate DATETIME = '01/01/2099' --End Value of Date Range

--Temporary Variables To Hold the Values During Processing of Each Date of Year
DECLARE
	@DayOfWeekInMonth INT,
	@DayOfWeekInYear INT,
	@DayOfQuarter INT,
	@WeekOfMonth INT,
	@CurrentYear INT,
	@CurrentMonth INT,
	@CurrentQuarter INT

/*Table Data type to store the day of week count for the month and year*/
DECLARE @DayOfWeek TABLE (DOW INT, MonthCount INT, QuarterCount INT, YearCount INT)

INSERT INTO @DayOfWeek VALUES (1, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (2, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (3, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (4, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (5, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (6, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (7, 0, 0, 0)

--Extract and assign various parts of Values from Current Date to Variable

DECLARE @CurrentDate AS DATETIME = @StartDate
SET @CurrentMonth = DATEPART(MM, @CurrentDate)
SET @CurrentYear = DATEPART(YY, @CurrentDate)
SET @CurrentQuarter = DATEPART(QQ, @CurrentDate)

/*********************************************************************************
**/
--Proceed only if Start Date(Current date ) is less than End date you specified above

WHILE @CurrentDate < @EndDate
BEGIN

/*Begin day of week logic*/

     /*Check for Change in Month of the Current date if Month changed then
      Change variable value*/
	IF @CurrentMonth != DATEPART(MM, @CurrentDate)
	BEGIN
		UPDATE @DayOfWeek
		SET MonthCount = 0
		SET @CurrentMonth = DATEPART(MM, @CurrentDate)
	END

    /* Check for Change in Quarter of the Current date if Quarter changed then change
     Variable value*/
```

```
        IF @CurrentQuarter != DATEPART(QQ, @CurrentDate)
        BEGIN
                UPDATE @DayOfWeek
                SET QuarterCount = 0
                SET @CurrentQuarter = DATEPART(QQ, @CurrentDate)
        END

    /* Check for Change in Year of the Current date if Year changed then change
     Variable value*/


        IF @CurrentYear != DATEPART(YY, @CurrentDate)
        BEGIN
                UPDATE @DayOfWeek
                SET YearCount = 0
                SET @CurrentYear = DATEPART(YY, @CurrentDate)
        END

    -- Set values in table data type created above from variables

        UPDATE @DayOfWeek
        SET
                MonthCount = MonthCount + 1,
                QuarterCount = QuarterCount + 1,
                YearCount = YearCount + 1
        WHERE DOW = DATEPART(DW, @CurrentDate)

        SELECT
                @DayOfWeekInMonth = MonthCount,
                @DayOfQuarter = QuarterCount,
                @DayOfWeekInYear = YearCount
        FROM @DayOfWeek
        WHERE DOW = DATEPART(DW, @CurrentDate)

/*End day of week logic*/


/* Populate Your Dimension Table with values*/

        INSERT INTO [dbo].[DimDate]
        SELECT

                CONVERT (char(8),@CurrentDate,112) as DateKey,
                @CurrentDate AS Date,
                CONVERT (char(10),@CurrentDate,103) as FullDateUK,
                CONVERT (char(10),@CurrentDate,101) as FullDateUSA,
                DATEPART(DD, @CurrentDate) AS DayOfMonth,
                --Apply Suffix values like 1st, 2nd 3rd etc..
                CASE
                        WHEN DATEPART(DD,@CurrentDate) IN (11,12,13)
                        THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'th'
                        WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 1
```

```
            THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'st'
            WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 2
            THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'nd'
            WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 3
            THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'rd'
            ELSE CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'th'
            END AS DaySuffix,

DATENAME(DW, @CurrentDate) AS DayName,
DATEPART(DW, @CurrentDate) AS DayOfWeekUSA,

-- check for day of week as Per US and change it as per UK format
CASE DATEPART(DW, @CurrentDate)
            WHEN 1 THEN 7
            WHEN 2 THEN 1
            WHEN 3 THEN 2
            WHEN 4 THEN 3
            WHEN 5 THEN 4
            WHEN 6 THEN 5
            WHEN 7 THEN 6
            END
            AS DayOfWeekUK,

@DayOfWeekInMonth AS DayOfWeekInMonth,
@DayOfWeekInYear AS DayOfWeekInYear,
@DayOfQuarter AS DayOfQuarter,
DATEPART(DY, @CurrentDate) AS DayOfYear,
DATEPART(WW, @CurrentDate) + 1 - DATEPART(WW, CONVERT(VARCHAR,
DATEPART(MM, @CurrentDate)) + '/1/' + CONVERT(VARCHAR,
DATEPART(YY, @CurrentDate))) AS WeekOfMonth,
(DATEDIFF(DD, DATEADD(QQ, DATEDIFF(QQ, 0, @CurrentDate), 0),
@CurrentDate) / 7) + 1 AS WeekOfQuarter,
DATEPART(WW, @CurrentDate) AS WeekOfYear,
DATEPART(MM, @CurrentDate) AS Month,
DATENAME(MM, @CurrentDate) AS MonthName,
CASE
            WHEN DATEPART(MM, @CurrentDate) IN (1, 4, 7, 10) THEN 1
            WHEN DATEPART(MM, @CurrentDate) IN (2, 5, 8, 11) THEN 2
            WHEN DATEPART(MM, @CurrentDate) IN (3, 6, 9, 12) THEN 3
            END AS MonthOfQuarter,
DATEPART(QQ, @CurrentDate) AS Quarter,
CASE DATEPART(QQ, @CurrentDate)
            WHEN 1 THEN 'First'
            WHEN 2 THEN 'Second'
            WHEN 3 THEN 'Third'
            WHEN 4 THEN 'Fourth'
            END AS QuarterName,
DATEPART(YEAR, @CurrentDate) AS Year,
'CY ' + CONVERT(VARCHAR, DATEPART(YEAR, @CurrentDate)) AS YearName,
LEFT(DATENAME(MM, @CurrentDate), 3) + '-' + CONVERT(VARCHAR,
DATEPART(YY, @CurrentDate)) AS MonthYear,
RIGHT('0' + CONVERT(VARCHAR, DATEPART(MM, @CurrentDate)),2) +
```

```
                    CONVERT(VARCHAR, DATEPART(YY, @CurrentDate)) AS MMYYYY,
                    CONVERT(DATETIME, CONVERT(DATE, DATEADD(DD, - (DATEPART(DD,
                    @CurrentDate) - 1), @CurrentDate))) AS FirstDayOfMonth,
                    CONVERT(DATETIME, CONVERT(DATE, DATEADD(DD, - (DATEPART(DD,
                    (DATEADD(MM, 1, @CurrentDate)))), DATEADD(MM, 1,
                    @CurrentDate)))) AS LastDayOfMonth,
                    DATEADD(QQ, DATEDIFF(QQ, 0, @CurrentDate), 0) AS FirstDayOfQuarter,
                    DATEADD(QQ, DATEDIFF(QQ, -1, @CurrentDate), -1) AS LastDayOfQuarter,
                    CONVERT(DATETIME, '01/01/' + CONVERT(VARCHAR, DATEPART(YY,
                    @CurrentDate))) AS FirstDayOfYear,
                    CONVERT(DATETIME, '12/31/' + CONVERT(VARCHAR, DATEPART(YY,
                    @CurrentDate))) AS LastDayOfYear,
                    NULL AS IsHolidaySL,
                    CASE DATEPART(DW, @CurrentDate)
                              WHEN 1 THEN 0
                              WHEN 2 THEN 1
                              WHEN 3 THEN 1
                              WHEN 4 THEN 1
                              WHEN 5 THEN 1
                              WHEN 6 THEN 1
                              WHEN 7 THEN 0
                              END AS IsWeekday,
                    NULL AS HolidaySL, (case when @CurrentDate = convert(date, sysdatetime()) then 1 else 0
end), 0, 0

          SET @CurrentDate = DATEADD(DD, 1, @CurrentDate)
END
```

ETL Development- Accumulating Fact Tables



1. Data was extracted from donations staging
2. Sort Donations according to the donations ID
3. Data was extracted from factCompletetime staging

4. Sort Donations according to the donations ID
5. All the data from donations staging table and process complete time from factCompletetime staging was merged.
6. Map the Donation Received date with the surrogate key of the date dimension using a lookup
7. Map the Resource Received date with the surrogate key of the date dimension using a lookup
8. Map the Rating ID with the surrogate key of the rating dimension using a lookup
9. Map the Donors ID with the surrogate key of the donors dimension using a lookup
10. Derive Insert and Modified date values using the GETDATE() function
11. Derive the txn_process_time_hours by finding the difference between accm_txn_complete_time and accm_txn_create_time in hours
12. The date is loaded into the FactDonations Dimension

Execution of Test Cases and TSR

Test Cases were generated for the two stages mentioned below:

1. Extracting data from Source to Staging tables
2. Extracting data from Staging to Dimension tables

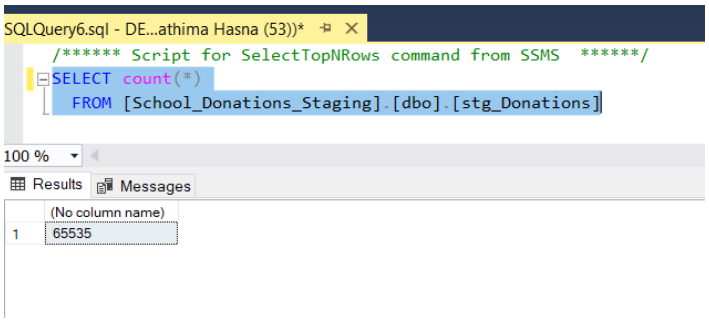
These test cases are generated to ensure that there is no data loss while loading, and to make sure that the data is of good quality.

Execution of Test Cases – Loading into Staging Tables

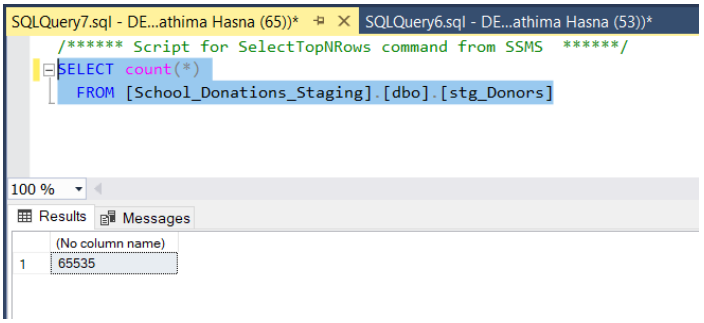
Test Scenario ID			1			
Test Case Description			Check for the count when transforming data from source to staging tables			
Pre-Requisite			Data should be loaded into the Staging tables in SQL tool			
S.No	**Action**	**SQL Query**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Check for the count when transforming data from Donations source to	`SELECT count(*) FROM [School_Donations_Staging].[dbo].[stg_Donations]`	65535	65535	Pass	Source Count and Staging Count of Donations data

	Donations Staging					are equal Refer 1.1 attachment
2	Check for the count when transforming data from Donors source to Donors Staging	`SELECT count(*) FROM [School_Donations_Staging].[dbo].[stg_Donors]`	65535	65535	Pass	Source Count and Staging Count of Donors data are equal Refer 1.2 attachment
3	Check for the count when transforming data from Projects source to Projects Staging	`SELECT count(*) FROM [School_Donations_Staging].[dbo].[stg_Projects]`	50	50	Pass	Source Count and Staging Count of Projects data are equal Refer 1.3 attachment
4	Check for the count when transforming data from Rating source to Rating Staging	`SELECT count(*) FROM [School_Donations_Staging].[dbo].[stg_Rating]`	17805	17805	Pass	Source Count and Staging Count of Rating data are equal Refer 1.4 attachment
5	Check for the count when transforming data	`SELECT count(*) FROM [School_Donations_Staging].[dbo].[stg_Schools]`	17805	17805	Pass	Source Count and Staging Count of

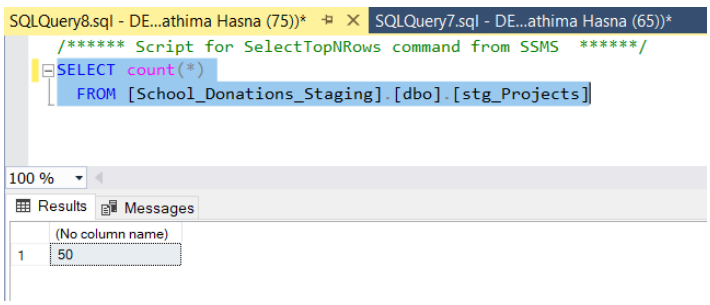
	from Schools source to Schools Staging					Schools data are equal Refer 1.5 attachment
6	Check for the count when transforming data from Teachers source to Teachers Staging	SELECT count(*) FROM [School_Donations_Staging].[dbo].[stg_Teachers]	49268	49268	Pass	Source Count and Staging Count of Teachers data are equal Refer 1.6 attachment



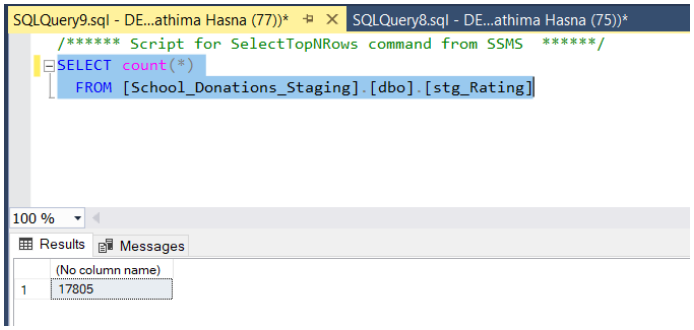
Attachment 1.1



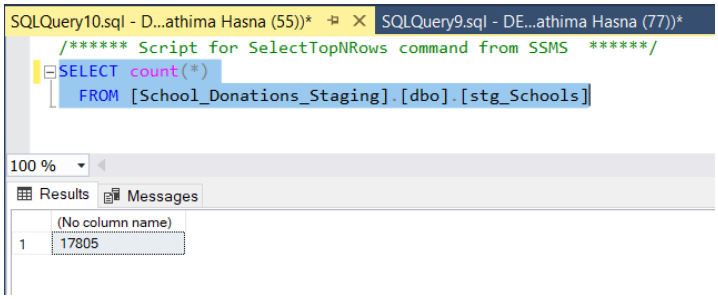
Attachment 1.2



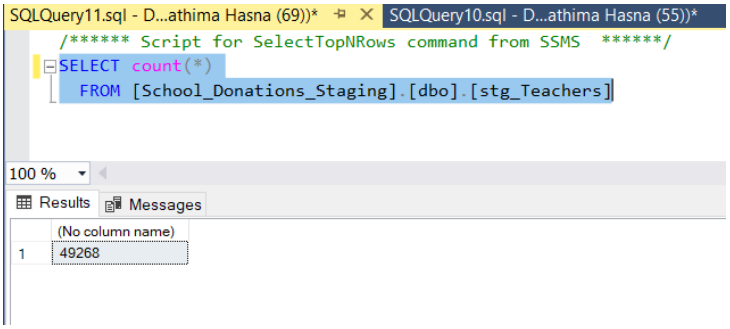
Attachment 1.3



Attachment 1.4



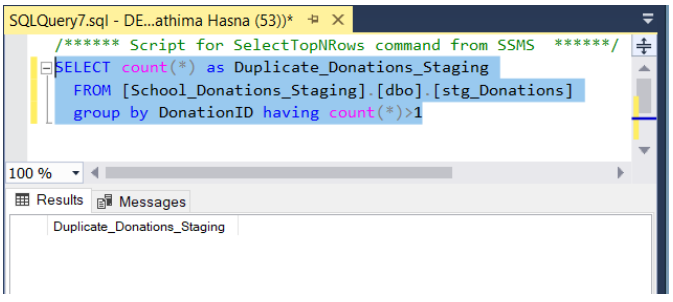
Attachment 1.5



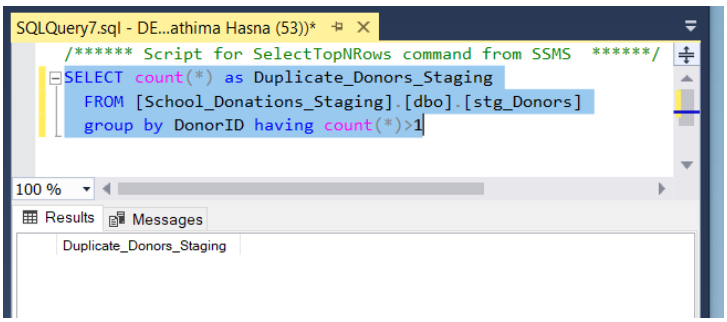
Attachment 1.6

Test Scenario ID			2			
Test Case Description			Check for duplicate values in staging tables			
Pre-Requisite			Data should be loaded into the Staging tables in SQL tool			
S.No	**Action**	**SQL Query**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Check whether the data has got duplicated in Donations Staging	`SELECT count(*) as Duplicate_Donations_Staging FROM [School_Donations_Staging].[dbo].[stg_Donations] group by DonationID having count(*)>1`	0	0	Pass	Data has not got duplicated in donations staging Refer 2.1 attachment
2	Check whether the data has got duplicated in Donors Staging	`SELECT count(*) as Duplicate_Donors_Staging FROM [School_Donations_Staging].[dbo].[stg_Donors] group by DonorID having count(*)>1`	0	0	Pass	Data has not got duplicated in Donors staging Refer 2.2 attachment
3	Check whether the data has got duplicated in	`SELECT count(*) as Duplicate_Projects_Staging FROM [School_Donations_Staging].[dbo].[stg_Projects] group by ProjectID having count(*)>1`	0	0	Pass	Data has not got duplicated in Projects staging

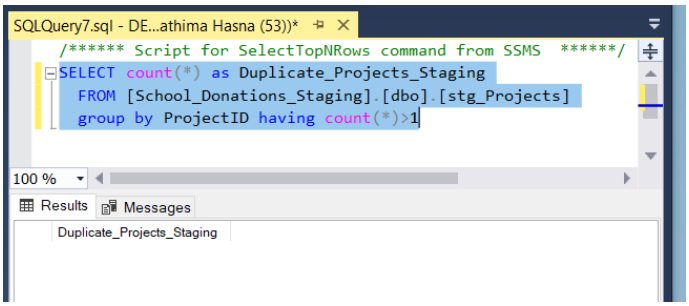
	Projects Staging					Refer 2.3 attachment
4	Check whether the data has got duplicated in Rating Staging	SELECT count(*) as Duplicate_Rating_Staging FROM [School_Donations_Staging].[dbo].[stg_Rating] group by Rating_ID having count(*)>1	0	0	Pass	Data has not got duplicated in Rating staging Refer 2.4 attachment
5	Check whether the data has got duplicated in Schools Staging	SELECT count(*) as Duplicate_Schools_Staging FROM [School_Donations_Staging].[dbo].[stg_Schools] group by SchoolID having count(*)>1	0	0	Pass	Data has not got duplicated in Schools staging Refer 2.5 attachment
6	Check whether the data has got duplicated in Teachers Staging	SELECT count(*) as Duplicate_Teachers_Staging FROM [School_Donations_Staging].[dbo].[stg_Teachers] group by TeacherID having count(*)>1	0	0	Pass	Data has not got duplicated in Teachers staging Refer 2.6 attachment



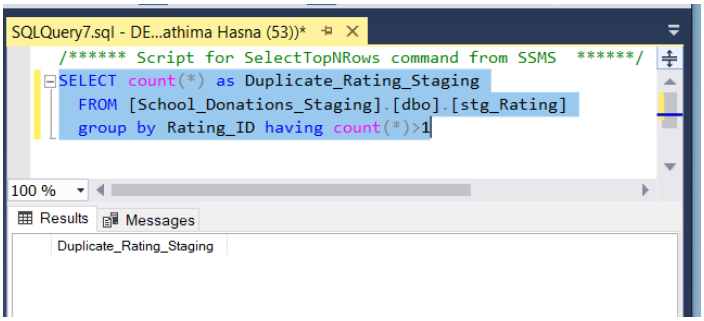
Attachment 2.1



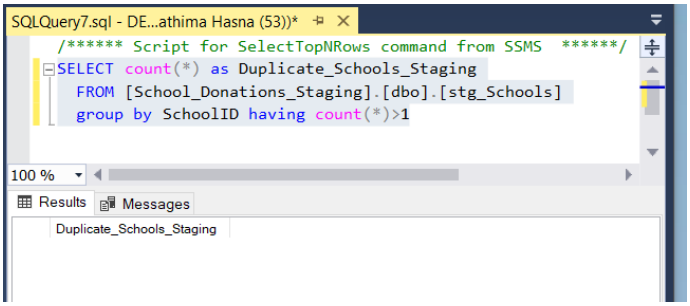
Attachment 2.2



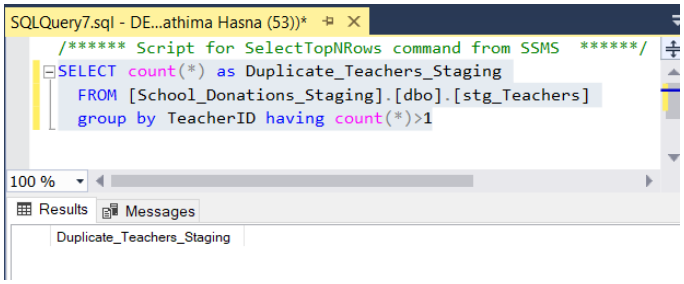
Attachment 2.3



Attachment 2.4



Attachment 2.5

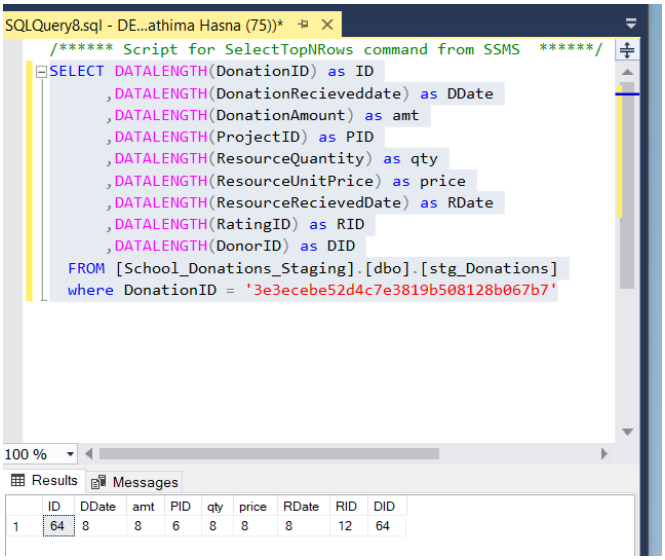


Attachment 2.6

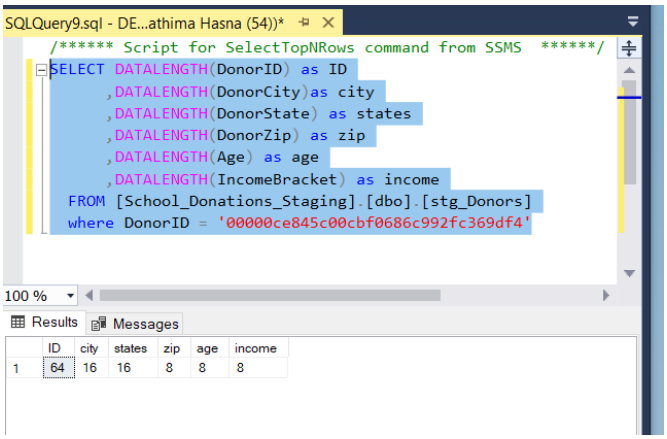
Test Scenario ID			3			
Test Case Description			Check for data length in staging tables			
Pre-Requisite			Data should be loaded into the Staging tables in SQL tool			
S.No	**Action**	**SQL Query**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Check whether the data length in Donations Source table and Donations Staging table are equal	SELECT DATALENGTH(DonationID) as ID ,DATALENGTH(DonationRecieveddate) as DDate ,DATALENGTH(DonationAmount) as amt ,DATALENGTH(ProjectID) as PID ,DATALENGTH(ResourceQuantity) as qty ,DATALENGTH(ResourceUnitPrice) as price ,DATALENGTH(ResourceRecievedDate) as RDate ,DATALENGTH(RatingID) as RID ,DATALENGTH(DonorID) as DID	ID – 64 DDate - 8 Amt – 8 PID – 6 Qty – 8 Price – 8	ID – 64 DDate - 8 Amt – 8 PID – 6 Qty – 8 Price – 8	Pass	Data lengths are equal Refer 3.1 attachment

		FROM [School_Donations_Staging].[dbo].[stg_Donations] where DonationID = '3e3ecebe52d4c7e3819b508128b067b7'	RDate -8 RID – 12 DID - 64	RDate -8 RID – 12 DID - 64		
2	Check whether the data length in Donors Source table and Donors Staging table are equal	SELECT DATALENGTH(DonorID) as ID ,DATALENGTH(DonorCity)as city ,DATALENGTH(DonorState) as states ,DATALENGTH(DonorZip) as zip ,DATALENGTH(Age) as age ,DATALENGTH(IncomeBracket) as income FROM [School_Donations_Staging].[dbo].[stg_Donors] where DonorID = '00000ce845c00cbf0686c992fc369df4'	ID – 64 City – 16 States – 16 Zip – 8 Age – 8 Incom e - 8	ID – 64 City – 16 States – 16 Zip – 8 Age – 8 Incom e - 8	Pass	Data lengths are equal Refer 3.2 attachm ent
3	Check whether the data length in Projects Source table and Projects Staging table are equal	SELECT DATALENGTH(ProjectID) as PID ,DATALENGTH(TeachersID) as TID ,DATALENGTH(ProjectName) as name ,DATALENGTH(ProjectBudget) as budget FROM [School_Donations_Staging].[dbo].[stg_Projects] where ProjectID = 'P1'	PID – 4 TID – 64 Name – 36 Budge t - 8	PID – 4 TID – 64 Name – 36 Budge t - 8	Pass	Data lengths are equal Refer 3.3 attachm ent
4	Check whether the data length in Rating Source table and Rating Staging table are equal	SELECT DATALENGTH(Rating_ID) as ID ,DATALENGTH(Stars) as stars ,DATALENGTH(School_ID) as SID ,DATALENGTH(Reveiwed_by) as rby ,DATALENGTH(Reviewed_Date) as RDate FROM [School_Donations_Staging].[dbo].[stg_Rating] where Rating_ID = 'R1'	ID – 4 Star – 8 SID – 64 Rby – 18 RDate - 8	ID – 4 Star – 8 SID – 64 Rby – 18 RDate - 8	Pass	Data lengths are equal Refer 3.4 attachm ent
5	Check whether the data length in Schools Source table and Schools	SELECT DATALENGTH(SchoolID) as ID ,DATALENGTH(SchoolName) as name ,DATALENGTH(SchoolMetroType) as stype ,DATALENGTH(SchoolState) as states ,DATALENGTH(SchoolZip) as zip ,DATALENGTH(SchoolCity) as city ,DATALENGTH(SchoolCounty) as country	ID – 64 Name – 52 Stype – 10 States – 26	ID – 64 Name – 52 Stype – 10 States – 26	Pass	Data lengths are equal Refer 3.5 attachm ent

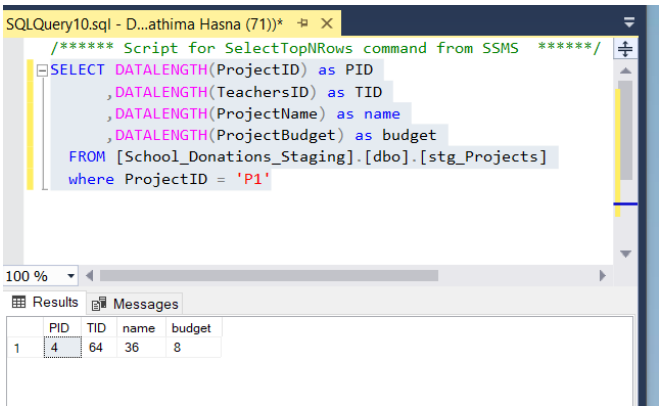
	Staging table are equal	,DATALENGTH(SchoolDistrict) as district FROM [School_Donations_Staging].[dbo].[stg_Schools] where SchoolID = '00003e0fdd601b8ea0a6eb44057b9c5e'	Zip – 8 City – 24 Country – 18 District - 56	Zip – 8 City – 24 Country – 18 District - 56		
6	Check whether the data length in Teachers Source table and Teachers Staging table are equal	SELECT DATALENGTH(TeacherID) as ID ,DATALENGTH(TeacherPrefix) as prefix ,DATALENGTH(TeacherFirstProjectPostedDate) as TPDate FROM [School_Donations_Staging].[dbo].[stg_Teachers] where TeacherID = '018f955b593f2bca4d6adba696757a33'	ID – 32 Prefix – 3 TPDate - 8	ID – 32 Prefix – 3 TPDate - 8	Pass	Data lengths are equal Refer 3.6 attachment



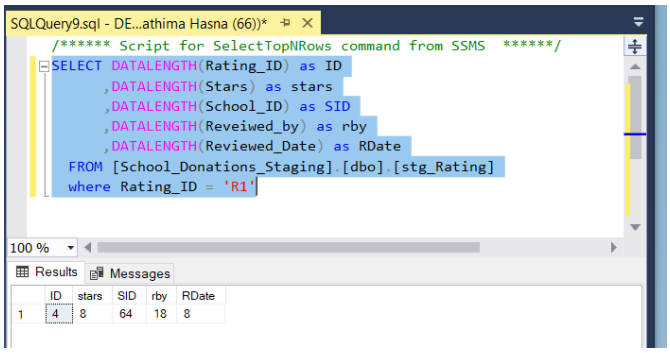
Attachment 3.1



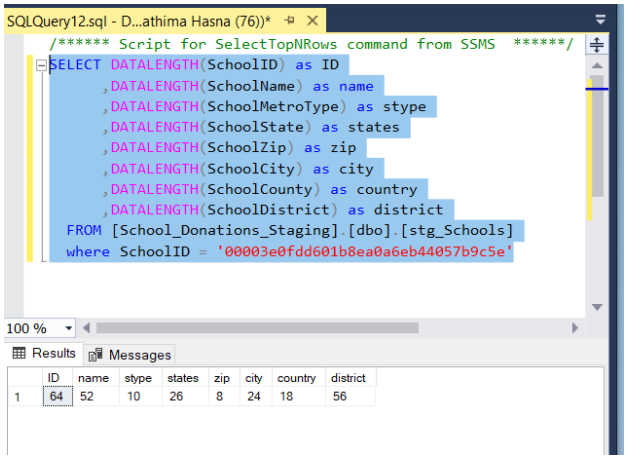
Attachment 3.2



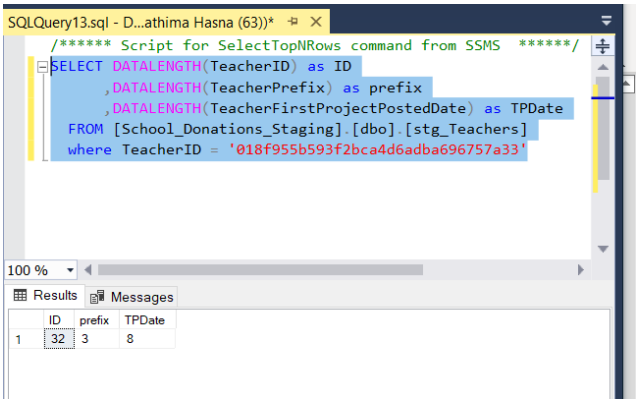
Attachment 3.3



Attachment 3.4



Attachment 3.5

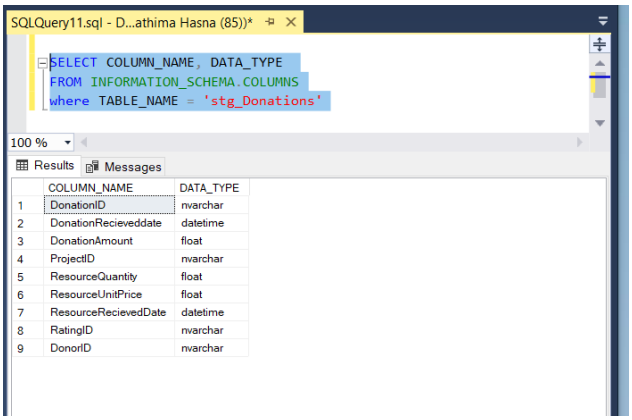


Attachment 3.6

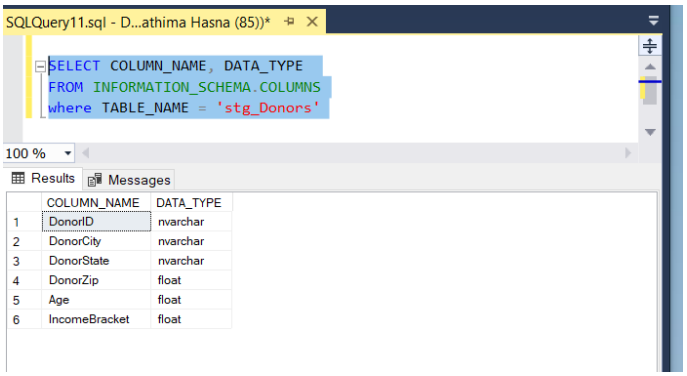
Test Scenario ID			4			
Test Case Description			Check for data type in staging tables			
Pre-Requisite			Data should be loaded into the Staging tables in SQL tool			
S.No	**Action**	**SQL Query**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Check whether the data types in Donations Source table and Donations	`SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'stg_Donations'`	DonationID – nvarchar DonationRecievedDate – datetime DonationAmount – float	DonationID – nvarchar DonationRecievedDate – datetime DonationAmount – float	Pass	Data types are same Refer 4.1 attachment

	Staging table are the same		ProjectID – nvarchar ResourceQuantity – float ResourceUnitPrice – float ResourceRecievedDate – datetime RatingID – nvarchar DonorID - nvarchar	ProjectID – nvarchar ResourceQuantity – float ResourceUnitPrice – float ResourceRecievedDate – datetime RatingID – nvarchar DonorID - nvarchar		
2	Check whether the data types in Donors Source table and Donors Staging table are equal	`SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'stg_Donors'`	DonorID – nvarchar DonorCity – nvarchar DonorState – nvarchar DonorZip – float Age – float IncomeBracket - float	DonorID – nvarchar DonorCity – nvarchar DonorState – nvarchar DonorZip – float Age – float IncomeBracket - float	Pass	Data types are same Refer 4.2 attachment
3	Check whether the data types in Projects Source table and Projects Staging table are equal	`SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'stg_Projects'`	ProjectID – nvarchar TeachersID – navarchar ProjectName – nvarchar ProjectBudget - float	ProjectID – nvarchar TeachersID – navarchar ProjectName – nvarchar ProjectBudget - float	Pass	Data types are same Refer 4.3 attachment
4	Check whether the data types in Rating Source table and Rating Staging table are equal	`SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'stg_Rating'`	SchoolID – nvarchar Stars – float Rating_Id – nvarchar Reviewed_by – nvarchar Reviewed_Date - datetime	SchoolID – nvarchar Stars – float Rating_Id – nvarchar Reviewed_by – nvarchar Reviewed_Date - datetime	Pass	Data types are same Refer 4.4 attachment

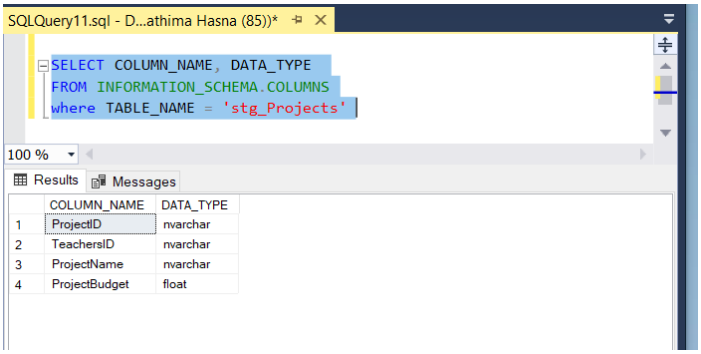
5	Check whether the data types in Schools Source table and Schools Staging table are equal	SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'stg_Schools'	SchoolID – nvarchar SchoolName – nvarchar SchoolMetroType - nvarchar SchoolState – nvarchar SchoolZip – float SchoolCity – nvarchar SchoolCountry – nvarchar SchoolDistrict - nvarchar	SchoolID – nvarchar SchoolName – nvarchar SchoolMetroType - nvarchar SchoolState – nvarchar SchoolZip – float SchoolCity – nvarchar SchoolCountry – nvarchar SchoolDistrict - nvarchar	Pass	Data types are same Refer 4.5 attachment
6	Check whether the data types in Teachers Source table and Teachers Staging table are equal	SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'stg_Teachers'	TeacherID – varchar TeacherPrefix – varchar TeacherFirstProjectPostedDate - datetime	TeacherID – varchar TeacherPrefix – varchar TeacherFirstProjectPostedDate - datetime	Pass	Data types are same Refer 4.6 attachment



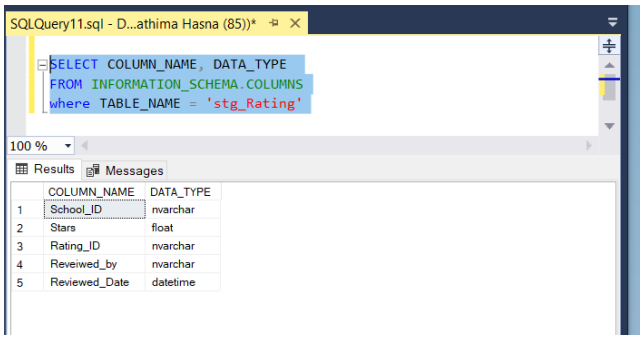
Attachment 4.1



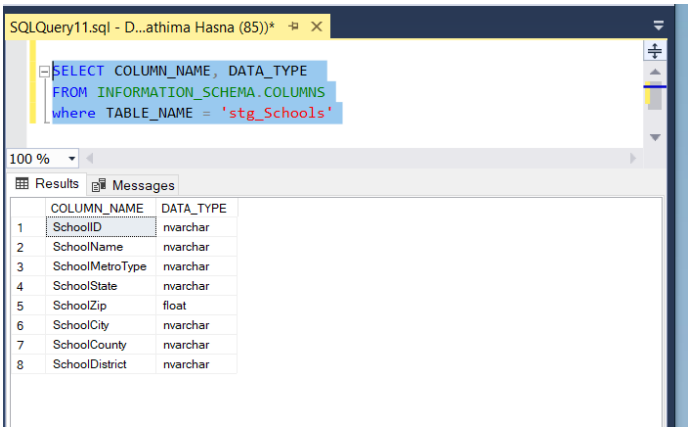
Attachment 4.2



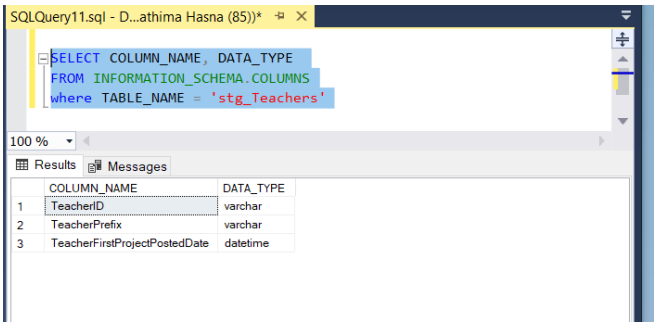
Attachment 4. 3



Attachment 4.4



Attachment 4.5



Attachment 4.6

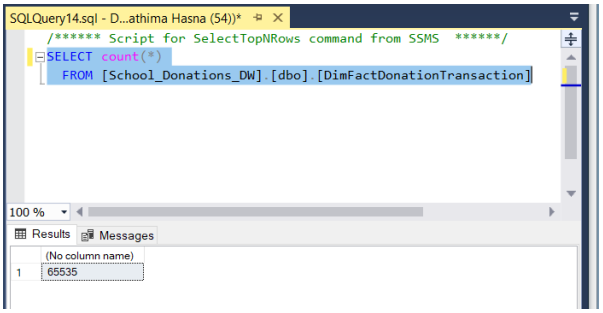
Execution of Test Cases – Loading into Dimension Tables

Test Scenario ID			5			
Test Case Description			Check for the count when transforming data from staging to dimension tables			
Pre-Requisite			Data should be loaded into the Dimension tables in SQL tool			
S.No	**Action**	**SQL Query**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Check for the count when	`SELECT count(*) FROM [School_Donations_DW].[dbo].[DimFactDonationTransaction]`	65535	65535	Pass	Staging Count

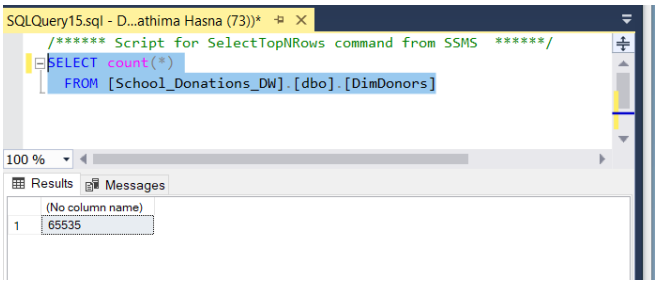
	transforming data from Donations Staging to Fact Donations Transaction Dimension					and Dimension Count of Fact Donations Transaction data are equal Refer 5.1 attachment
2	Check for the count when transforming data from Donors Staging to Donors dimension	`SELECT count(*) FROM [School_Donations_DW].[dbo].[DimDonors]`	65535	65535	Pass	Staging Count and Dimension Count of Donors data are equal Refer 5.2 attachment
3	Check for the count when transforming data from Projects Staging to Projects Dimension	`SELECT count(*) FROM [School_Donations_DW].[dbo].[DimProjects]`	50	50	Pass	Staging Count and Dimension Count of Projects data are equal Refer 5.3 attachment
4	Check for the count when transforming data from Rating Staging	`SELECT count(*) FROM [School_Donations_DW].[dbo].[DimRating]`	17805	17805	Pass	Staging Count and Dimension Count of

	to Rating Dimension					Rating data are equal Refer 5.4 attachment
5	Check for the count when transforming data from Schools Staging to School Dimension	SELECT count(*) FROM [School_Donations_DW].[dbo].[DimSchools]	17805	17805	Pass	Staging Count and Dimension Count of Schools data are equal Refer 5.5 attachment
6	Check for the count when transforming data from Teachers Staging to Teachers Dimension	SELECT count(*) FROM [School_Donations_DW].[dbo].[DimTeacher]	49268	49268	Pass	Staging Count and Dimension Count of Teachers data are equal Refer 5.6 attachment
7	Check for the count when transforming data from Donations Staging to Fact Donations Accumulating Dimension	SELECT count(*) FROM [School_Donations_DW].[dbo].[DimFactDonationsAccm]	65535	65535	Pass	Staging Count and Dimension Count of Fact Donations Accumulating data are equal

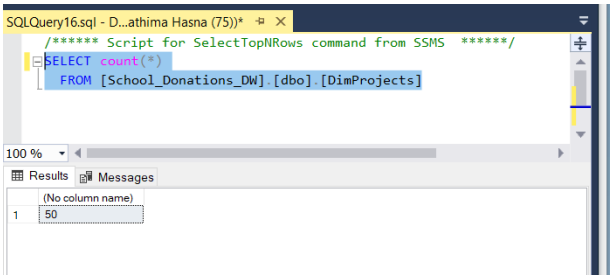
						Refer 5.7 attachment



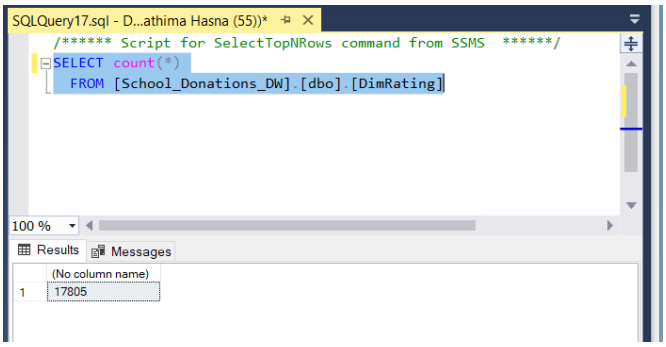
Attachment 5.2



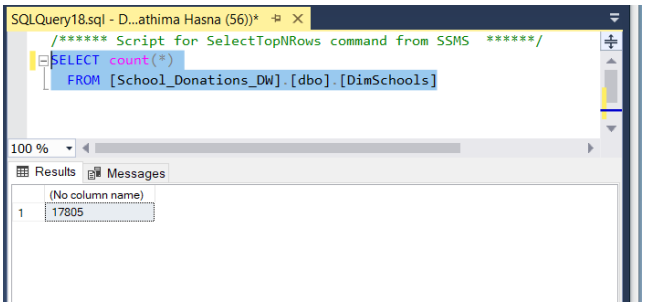
Attachment 5. 3



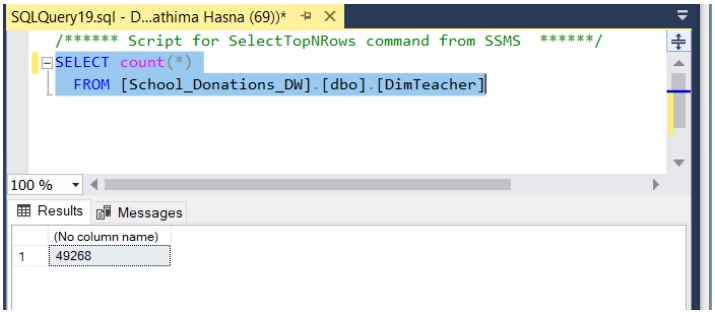
Attachment 5.3



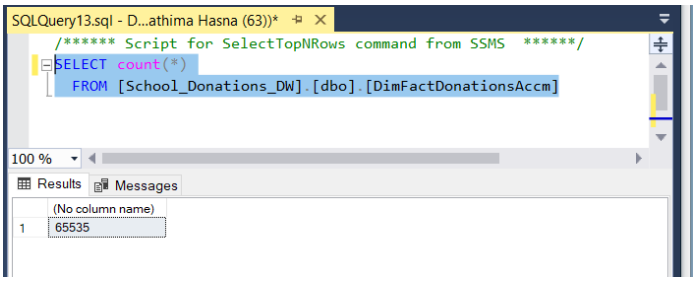
Attachment 5.4



Attachment 5.5



Attachment 5.6

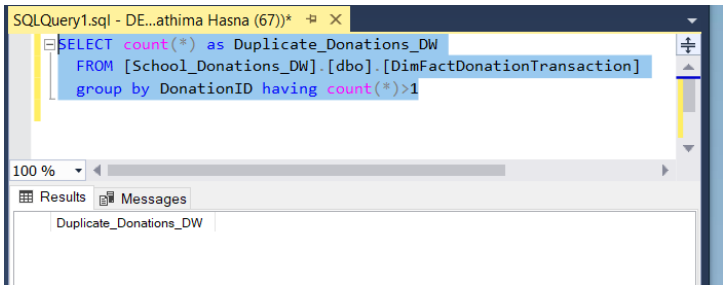


Attachment 5.7

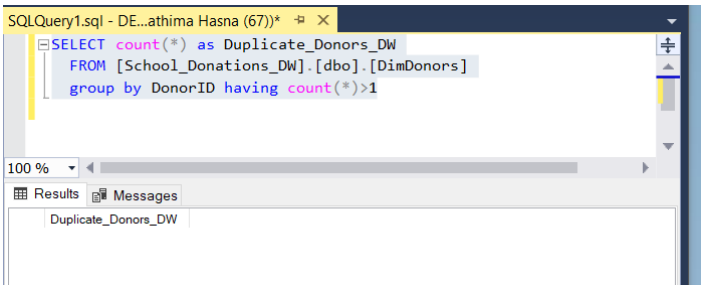
Test Scenario ID			6			
Test Case Description			Check for duplicate values in dimension tables			
Pre-Requisite			Data should be loaded into the Staging tables in SQL tool			
S.No	**Action**	**SQL Query**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Check whether the data has got duplicated in Fact Donation Transaction Dimension	`SELECT count(*) as Duplicate_Donations_DW FROM [School_Donations_DW].[dbo].[DimFactDonationTransaction] group by DonationID having count(*)>1`	0	0	Pass	Data has not got duplicated in Fact Donation Transaction Dimension Refer 6.1 attachment
2	Check whether the data has got duplicated in Donors Dimension	`SELECT count(*) as Duplicate_Donors_DW FROM [School_Donations_DW].[dbo].[DimDonors] group by DonorID having count(*)>1`	0	0	Pass	Data has not got duplicated in Donors Dimension Refer 6.2 attachment

3	Check whether the data has got duplicated in Projects Dimension	SELECT count(*) as Duplicate_Projects_DW FROM [School_Donations_DW].[dbo].[DimProjects] group by ProjectID having count(*)>1	0	0	Pass	Data has not got duplicated in Projects Dimension Refer 6.3 attachment
4	Check whether the data has got duplicated in Rating Dimension	SELECT count(*) as Duplicate_Rating_DW FROM [School_Donations_DW].[dbo].[DimRating] group by RatingID having count(*)>1	0	0	Pass	Data has not got duplicated in Rating Dimension Refer 6.4 attachment
5	Check whether the data has got duplicated in Schools Dimension	SELECT count(*) as Duplicate_Schools_DW FROM [School_Donations_DW].[dbo].[DimSchools] group by SchoolID having count(*)>1	0	0	Pass	Data has not got duplicated in Schools Dimension Refer 6.5 attachment
6	Check whether the data has got duplicated in Teachers Dimension	SELECT count(*) as Duplicate_Teachers_DW FROM [School_Donations_DW].[dbo].[DimTeacher] group by TeacherID having count(*)>1	0	0	Pass	Data has not got duplicated in Teachers Dimension Refer 6.6 attachment
7	Check whether the data	SELECT count(*) as Duplicate_Donations_DW	0	0	Pass	Data has not got duplicat

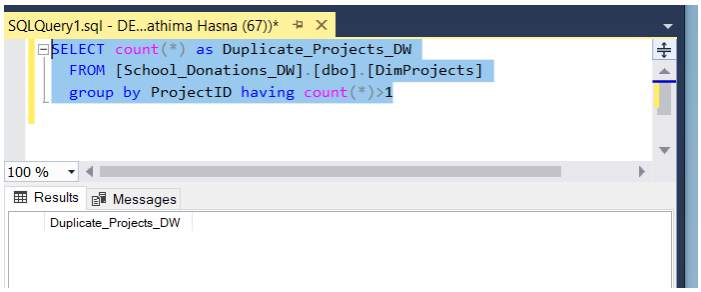
	has got duplicated in Fact Donation Accumulating Dimension	FROM [School_Donations_DW].[dbo].[DimFactDonationsAccm] group by DonationID having count(*)>1				ed in in Fact Donation Accumulating Dimension Refer 6.7 attachment



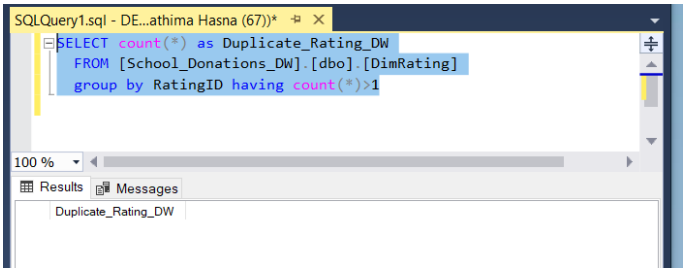
Attachment 61



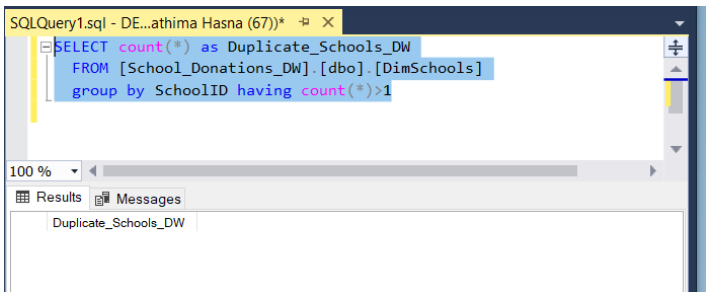
Attachment 6.2



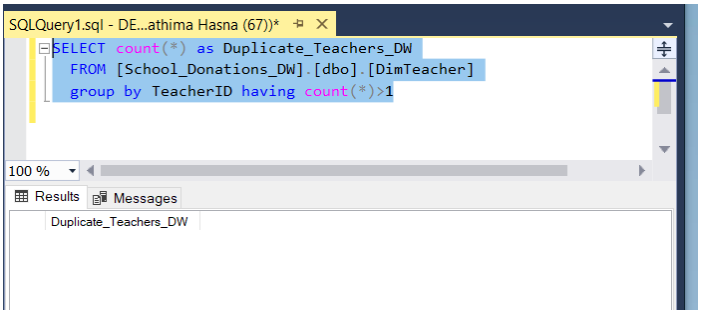
Attachment 6.3



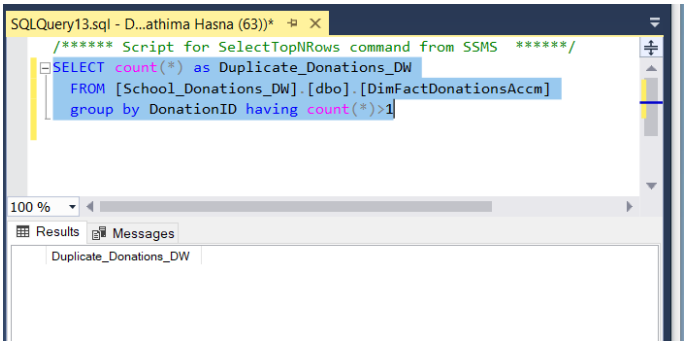
Attachment 6.4



Attachment 6.5



Attachment 6.6

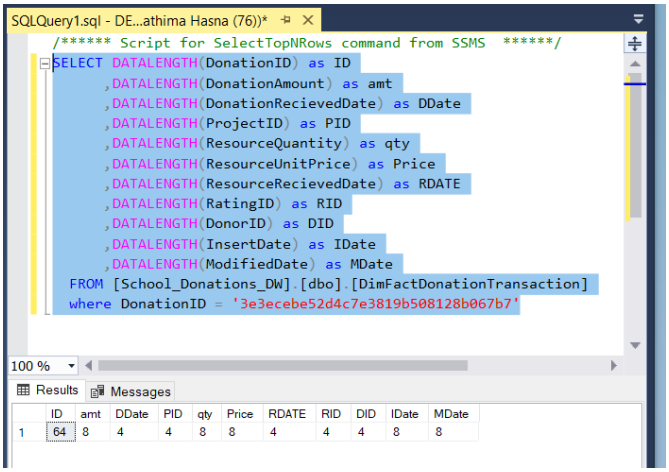


Attachment 6.7

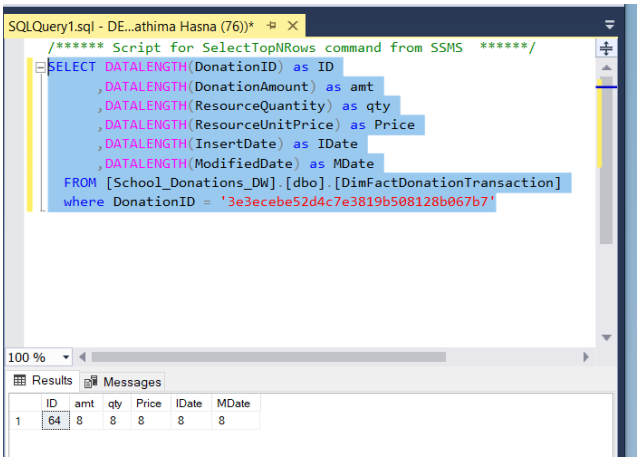
Test Scenario ID			7			
Test Case Description			Check for data length in dimension tables			
Pre-Requisite			Data should be loaded into the Dimension tables in SQL tool			
S.No	**Action**	**SQL Query**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Check whether the data length in Donations Staging table and Fact Donations Transaction Dimension table are equal	SELECT DATALENGTH(DonationID) as ID ,DATALENGTH(DonationAmount) as amt ,DATALENGTH(ResourceQuantity) as qty ,DATALENGTH(ResourceUnitPrice) as Price FROM [School_Donations_DW].[dbo].[DimFactDonationTransaction] where DonationID = '3e3ecebe52d4c7e3819b508128b067b7'	ID – 64 Amt – 8 Qty – 8 Price – 8	ID – 64 Amt – 8 Qty – 8 Price – 8	Pass	Data lengths are equal Refer 7.1 attachment
2	Check whether the data length in Donors Staging table and Donors Dimension table are equal	SELECT DATALENGTH(DonorID) as ID ,DATALENGTH(DonorCity) as city ,DATALENGTH(DonorState) as states ,DATALENGTH(DonorZip) as zip ,DATALENGTH(Age) as age ,DATALENGTH(IncomeBracket) as income FROM [School_Donations_DW].[dbo].[DimDonors] where DonorID = '00000ce845c00cbf0686c992fc369df4'	ID – 64 City – 16 States – 16 Zip – 8 Age – 8 Income - 8	ID – 64 City – 16 States – 16 Zip – 8 Age – 8 Income - 8	Pass	Data lengths are equal Refer 7.2 attachment
3	Check whether the data	SELECT DATALENGTH(ProjectID) as ID ,DATALENGTH(ProjectName) as Name ,DATALENGTH(ProjectBudget) as Budget	PID – 6	PID – 6	Pass	Data lengths

	length in Projects Staging table and Projects Dimension table are equal	FROM [School_Donations_DW].[dbo].[DimProjects] where ProjectID = 'P36'	Name – 38 Budget - 8	Name – 38 Budget - 8		are equal Refer 7.3 attachment
4	Check whether the data length in Rating Staging table and Rating Dimension table are equal	SELECT DATALENGTH(RatingID) as RID ,DATALENGTH(Reviewed_by) as rby ,DATALENGTH(Stars) as stars ,DATALENGTH(Reviewed_Date) as RDate FROM [School_Donations_DW].[dbo].[DimRating] where RatingID = 'R1'	ID – 4 Rby – 18 Stars – 5 RID – 4 RDate - 8	ID – 4 Rby – 18 Stars – 5 RID – 4 RDate - 8	Pass	Data lengths are equal Refer 7.4 attachment
5	Check whether the data length in Schools Staging table and Schools Dimension table are equal	SELECT DATALENGTH(SchoolID) as ID ,DATALENGTH(SchoolName) as name ,DATALENGTH(SchoolMetroType) as stype ,DATALENGTH(SchoolState) as States ,DATALENGTH(Schoolzip) as zip ,DATALENGTH(SchoolCity) as city ,DATALENGTH(SchoolCountry) as country ,DATALENGTH(SchoolDistrict) as district FROM [School_Donations_DW].[dbo].[DimSchools] where SchoolId = '00003e0fdd601b8ea0a6eb44057b9c5e'	ID – 64 Name – 52 Stype – 10 States – 26 Zip – 8 City – 24 Country – 18 District - 56	ID – 64 Name – 52 Stype – 10 States – 26 Zip – 8 City – 24 Country – 18 District - 56	Pass	Data lengths are equal Refer 7.5 attachment
6	Check whether the data length in Teachers Staging table and Teachers Dimension table are equal	/****** Script for SelectTopNRows command from SSMS ******/ SELECT DATALENGTH(TeacherID) as ID ,DATALENGTH(TeacherPrefix) as prefix ,DATALENGTH(TeacherPostedDate) as TPDate FROM [School_Donations_DW].[dbo].[DimTeacher] where TeacherID = '018f955b593f2bca4d6adba696757a33'	ID – 32 Prefix – 3 TPDate - 8	ID – 32 Prefix – 3 TPDate - 8	Pass	Data lengths are equal Refer 7.6 attachment

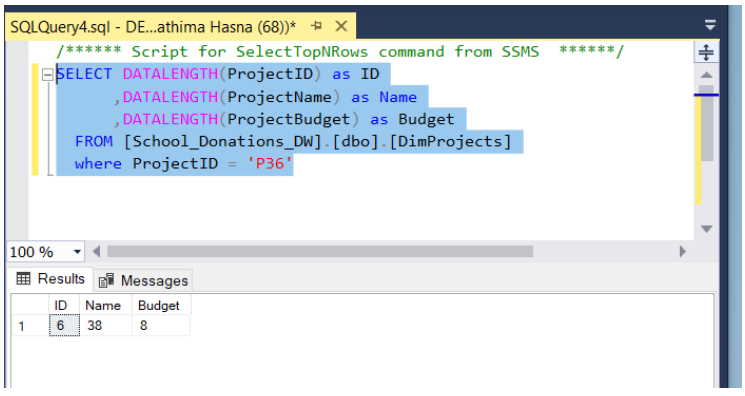
7	Check whether the data length in Donations Staging table and Fact Donations Accumulating Dimension table are equal	SELECT DATALENGTH(DonationID) as ID ,DATALENGTH(DonationAmount) as amt ,DATALENGTH(ResourceQuantity) as qty ,DATALENGTH(ResourceUnitPrice) as Price FROM [School_Donations_DW].[dbo].[DimFactDonationsAccm] where DonationID = '3e3ecebe52d4c7e3819b508128b067b7'	ID – 64 Amt – 8 Qty – 8 Price – 8	ID – 64 Amt – 8 Qty – 8 Price – 8	Pass	Data lengths are equal Refer 7.1 attachment



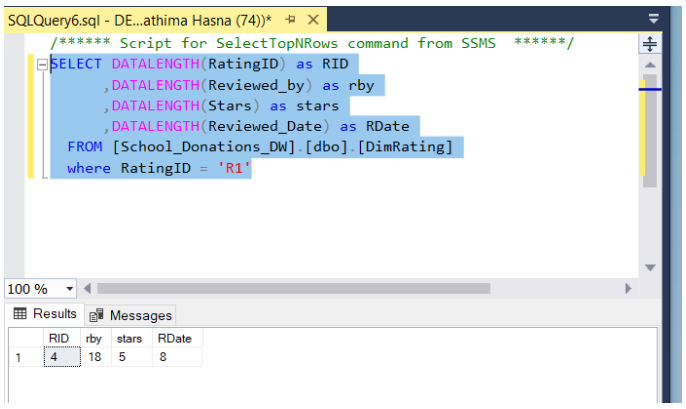
Attachment 7.1



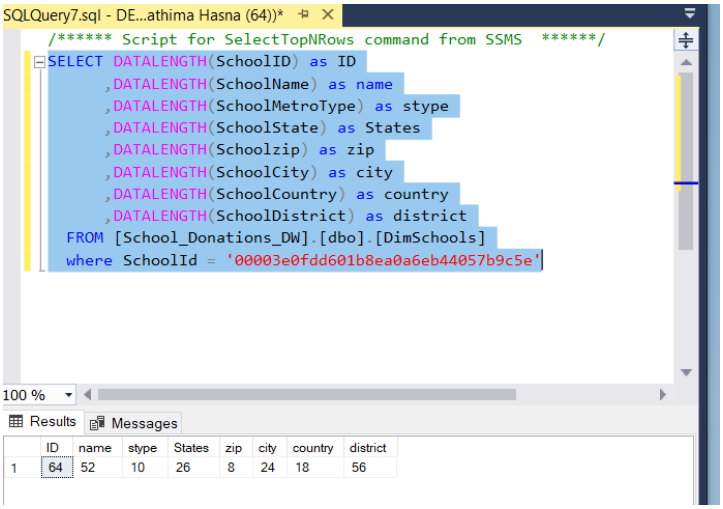
Attachment 7.2



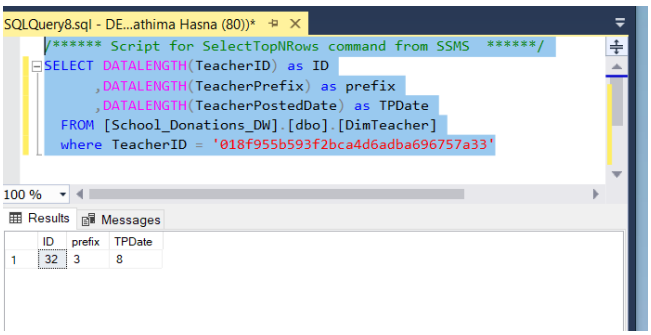
Attachment 7.3



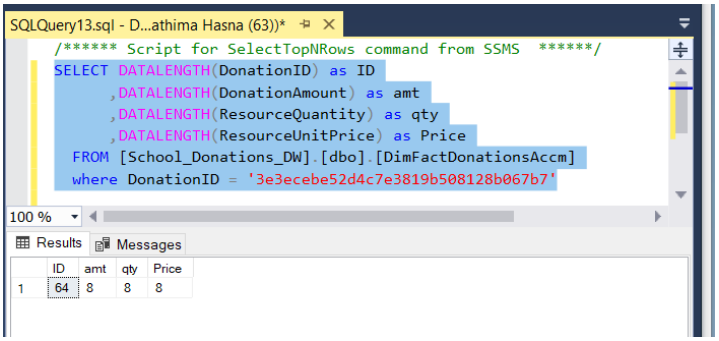
Attachment 7.4



Attachment 7.5



Attachment 7.6

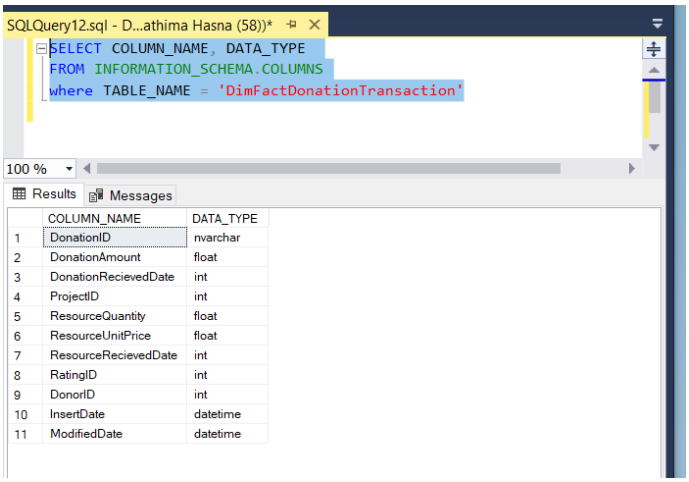


Attachment 7.6

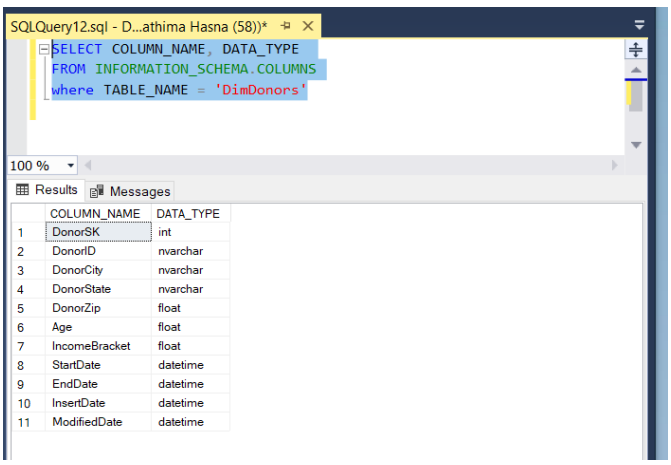
Test Scenario ID			8			
Test Case Description			Check for data type in staging tables			
Pre-Requisite			Data should be loaded into the Staging tables in SQL tool			
S.No	**Action**	**SQL Query**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Check whether the data types in Donations Staging table and	SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'DimFactDonationTransaction'	DonationID – nvarchar DonationAmount – float ResourceQuantity – float	DonationID – nvarchar DonationAmount – float ResourceQuantity – float	Pass	Data types are same Refer 8.1

	Fact Donations Transaction Dimension table are the same		ResourceUnitPrice – float	ResourceUnitPrice – float		attachment
2	Check whether the data types in Donors Staging table and Donors Dimension table are equal	`SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'DimDonors'`	DonorID – nvarchar DonorCity – nvarchar DonorState – nvarchar DonorZip – float Age – float IncomeBracket - float	DonorID – nvarchar DonorCity – nvarchar DonorState – nvarchar DonorZip – float Age – float IncomeBracket - float	Pass	Data types are same Refer 8.2 attachment
3	Check whether the data types in Projects Staging table and Projects Dimension table are equal	`SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'DimProjects'`	ProjectID – nvarchar ProjectName – nvarchar ProjectBudget - float	ProjectID – nvarchar ProjectName – nvarchar ProjectBudget - float	Pass	Data types are same Refer 8.3 attachment
4	Check whether the data types in Rating Staging table and Rating Dimension table are equal	`SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'DimRating'`	Rating_Id – nvarchar Stars – float Reviewed_by – nvarchar Reviewed_Date - datetime	Rating_Id – nvarchar Stars – float Reviewed_by – nvarchar Reviewed_Date - datetime	Pass	Data types are same Refer 8.4 attachment
5	Check whether the data types in Schools Stagingta	`SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'DimSchools'`	SchoolID – nvarchar SchoolName – nvarchar SchoolMetroType - nvarchar	SchoolID – nvarchar SchoolName – nvarchar SchoolMetroType - nvarchar	Pass	Data types are same Refer 8.5

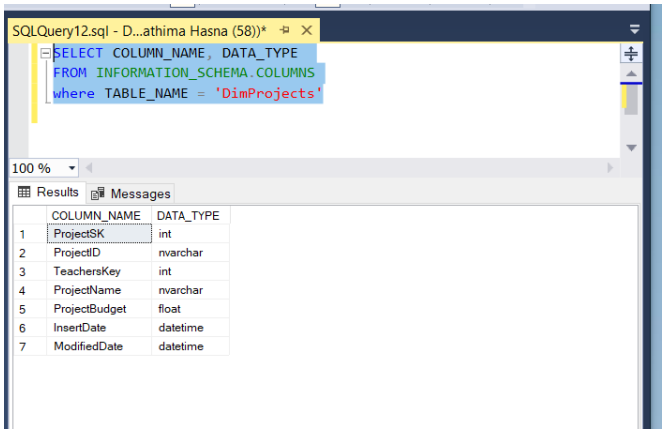
	ble and Schools Dimension table are equal		SchoolState – nvarchar SchoolZip – float SchoolCity – nvarchar SchoolCountry – nvarchar SchoolDistrict - nvarchar	SchoolState – nvarchar SchoolZip – float SchoolCity – nvarchar SchoolCountry – nvarchar SchoolDistrict - nvarchar		attachment
6	Check whether the data types in Teachers Staging table and Teachers Dimension table are equal	`SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'DimTeacher'`	TeacherID – varchar TeacherPrefix – varchar TeacherFirstProjectPostedDate - datetime	TeacherID – varchar TeacherPrefix – varchar TeacherFirstProjectPostedDate - datetime	Pass	Data types are same Refer 8.6 attachment
7	Check whether the data types in Donations Staging table and fact Donations Accumulating Dimension table are the same	`SELECT COLUMN_NAME, DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS where TABLE_NAME = 'DimFactDonationsAccm'`	DonationID – nvarchar DonationAmount – float ResourceQuantity – float ResourceUnitPrice – float	DonationID – nvarchar DonationAmount – float ResourceQuantity – float ResourceUnitPrice – float	Pass	Data types are same Refer 8.7 attachment



Attachment 8.1

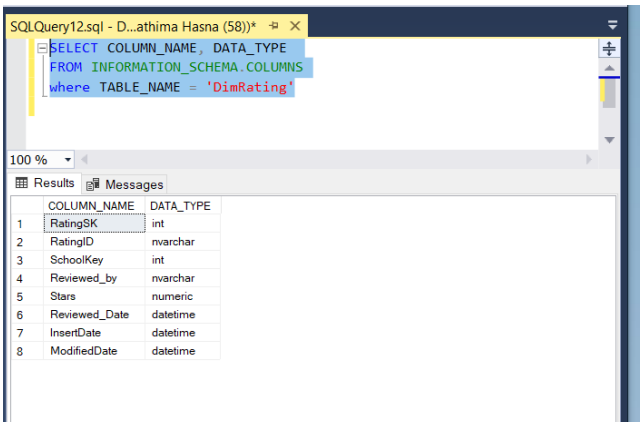


Attachment 8.2



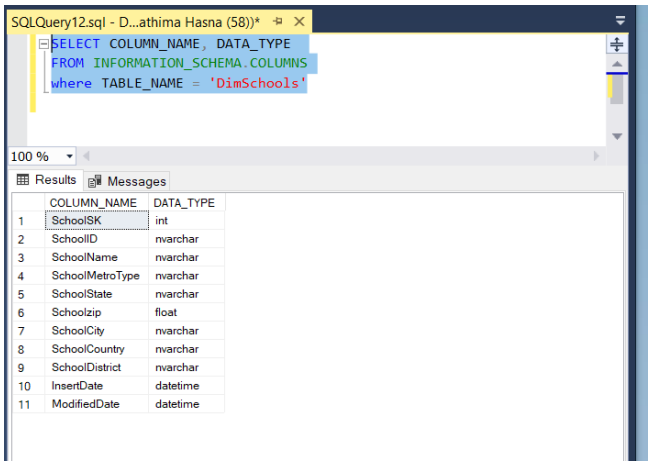
	COLUMN_NAME	DATA_TYPE
1	ProjectSK	int
2	ProjectID	nvarchar
3	TeachersKey	int
4	ProjectName	nvarchar
5	ProjectBudget	float
6	InsertDate	datetime
7	ModifiedDate	datetime

Attachment 8.3



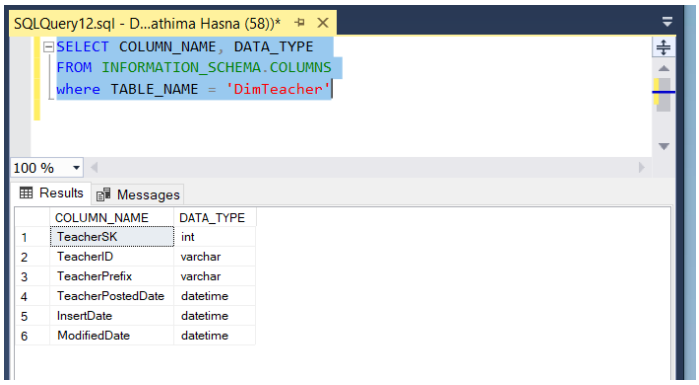
	COLUMN_NAME	DATA_TYPE
1	RatingSK	int
2	RatingID	nvarchar
3	SchoolKey	int
4	Reviewed_by	nvarchar
5	Stars	numeric
6	Reviewed_Date	datetime
7	InsertDate	datetime
8	ModifiedDate	datetime

Attachment 8.4



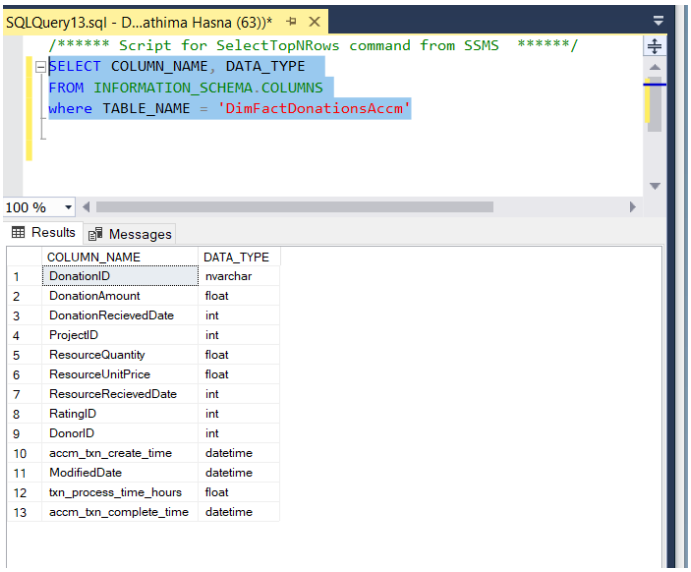
	COLUMN_NAME	DATA_TYPE
1	SchoolSK	int
2	SchoolID	nvarchar
3	SchoolName	nvarchar
4	SchoolMetroType	nvarchar
5	SchoolState	nvarchar
6	Schoolzip	float
7	SchoolCity	nvarchar
8	SchoolCountry	nvarchar
9	SchoolDistrict	nvarchar
10	InsertDate	datetime
11	ModifiedDate	datetime

Attachment 8.5



	COLUMN_NAME	DATA_TYPE
1	TeacherSK	int
2	TeacherID	varchar
3	TeacherPrefix	varchar
4	TeacherPostedDate	datetime
5	InsertDate	datetime
6	ModifiedDate	datetime

Attachment 8.6



	COLUMN_NAME	DATA_TYPE
1	DonationID	nvarchar
2	DonationAmount	float
3	DonationRecievedDate	int
4	ProjectID	int
5	ResourceQuantity	float
6	ResourceUnitPrice	float
7	ResourceRecievedDate	int
8	RatingID	int
9	DonorID	int
10	accm_txn_create_time	datetime
11	ModifiedDate	datetime
12	txn_process_time_hours	float
13	accm_txn_complete_time	datetime

Attachment 8.7

Test Summary Report

Please find the summary of the executed test cases below.

Cycle number	Number of test Cases	Passed	Failed	Comments
1	8	8	0	All the test cases were passed